UNIVERSITY MONTPELLIER 2

DOCTORAL SCHOOL I2S

SCIENCES ET TECHNOLOGIES DE L'INFORMATION

P H D T H E S I S

to obtain the title of

PhD of Science

of the University Montpellier

Speciality : Computer Science

Defended by

Chakkrit Preuksakarn

Reconstructing Plant Architecture from 3D Laser Scanner Data

prepared at Virtual Plants Team, UMR AGAP,
CIRAD/INRIA/INRA

defended on December 19, 2012

Jury :

Reviewers/President :	Bernard Mourrain	-	INRIA
Reviewers :	Frédéric Barret	-	INRA
Advisor :	Christophe Godin	-	INRIA
Co-Advisor :	Frédéric Boudon	-	CIRAD
Examinators :	Gerard Subsol	-	CNRS
	Geraldine Morin	-	INPT
Invited :	Pierre-Eric Lauri	-	INRA

Abstract

Virtual plant models can be visually realistic for computer graphics applications. However, in the context of biology and agronomy, acquisition of accurate models of real plants is still a tedious and time consuming task and is a major bottleneck for the construction of quantitative models of plant development.

Recently, 3D laser scanners have made it possible to acquire 3D images on which each pixel has an associate depth corresponding to the distance between the scanner and the pinpointed surface of the object. However, a plant is usually a set of discontinuous surfaces fuzzily distributed in a volume of vegetation. Classical geometrical reconstruction fails for this particular type of geometry.

In this thesis, we present a method for reconstructing virtual models of plants from laser scanning of real-world vegetation. Measuring plants with laser scanners produces data with different levels of precision. Points set are usually dense on the surface of the trunk and of the main branches, but only sparsely cover thin branches. The core of our method is to iteratively create the skeletal structure of the plant according to local density of point set. This is achieved thanks to a method locally adaptive to the levels of precision of the data that combine a contraction phase and a local point tracking algorithm.

In addition, we developed a quantitative evaluation procedure to compare our reconstructions against expertised structures of real plants. For this, we first explore the use of an edit distance between tree graphs. Alternatively, we formalize the comparison as an assignment problem to find the best matching between the two structures and quantify their differences.

Résumé

En infographie, les modèles virtuels de plantes sont de plus en plus réalistes visuellement. Cependant, dans le contexte de la biologie et l'agronomie, l'acquisition de modèles précis de plantes réelles reste un problème majeur pour la construction de modèles quantitatifs du développement des plantes.

Récemment, des scanners laser 3D permettent d'acquérir des images 3D avec pour chaque pixel une profondeur correspondant à la distance entre le scanner et la surface de l'objet visé. Cependant, une plante est généralement un ensemble important de petites surfaces sur lesquelles les méthodes classiques de reconstruction échouent.

Dans cette thèse, nous présentons une méthode pour reconstruire des modèles virtuels de plantes à partir de scans laser. Mesurer des plantes avec un scanner laser produit des données avec différents niveaux de précision. Les scans sont généralement denses sur la surface des branches principales mais recouvrent avec peu de points les branches fines. Le cur de notre méthode est de créer itérativement un squelette de la structure de la plante en fonction de la densité locale de points. Pour cela, une méthode localement adaptative a été développée qui combine une phase de contraction et un algorithme de suivi de points.

Nous présentons également une procédure d'évaluation quantitative pour comparer nos reconstructions avec des structures reconstruites par des experts de plantes réelles. Pour cela, nous explorons d'abord l'utilisation d'une distance d'édition entre arborescence. Finalement, nous formalisons la comparaison sous forme d'un problème d'assignation pour trouver le meilleur appariement entre deux structures et quantifier leurs différences.

Acknowledgements

First up, I would like to express my gratitude to my initial PhD supervisor, Christophe Godin. His guidance and support during this process have been invaluable. Next I would like to thank Frédéric Boudon, who took over the task of supervision and, more importantly, help this work possible. His enthusiasm and scientific knowledge have always been a great motivation and a driving force behind this thesis. This work has greatly benefited from his ideas and advice.

I extend sincere thanks to the members of my thesis committee. Thank you to Bernard Mourrain and Frédéric Barret for many insightful comments on the draft of this thesis and for his availability. Thank you to Gerard Subsol, Geraldine Morin, and Pierre-Eric Lauri for their constructive comments and discussions that we had on the defense.

I would also like to thank Pascal Ferraro and Jean-Beptiste Durand for helping with the design and the implementation of several algorithms developed in this thesis. In addition, my thanks go to Eero Nikinmaa and Eric Casella who help greatly with the equipment and data for testing.

Many thanks go to all my colleagues at the Virtual Plant Team. They have always created a very sociable and friendly atmosphere. Special thanks go to Yassin Refahi for all your help.

I would also like to acknowledge the generous financial support I received from the Kasetsart University, Thailand and the National Institute for Research in Computer Science and Control (INRIA).

Last, I would like also to thank the most important people in my life, my wife and my son for always being there for me and supporting me unconditionally.

Contents

Introduction

Over the last decade, plant modeling have become popular in computer graphics and related areas. It is not only major elements of virtual natural scenery in the graphic industry, but also give new opportunities to scientists to study the complex 3D architecture of plants. Understanding geometry of plants is a key factor for studying interaction between plants and environment (light, pest and disease propagation, etc.). These key benefits led the researcher to design digitizing method for generating accurate virtual plant models.

However, most of actual measurement methodologies are manual and extremely time consuming. This is a major issue in the reconstruction of quantitative models of plant development. With recent advances in laser scanning, direct captures of 3D data of plants has become possible. Such captures produce point clouds representing surface area of plants. The raw output is the set of spatial coordinates (x,y,z) of points seen by the camera at the surface of the plants. Therefore, most applicatifaons require the reconstruction of complete plant geometry from the captured point cloud. Although successful in most applications, such as for capturing archeological artifacts or urban geometry, this technology does not achieve acceptable results when applied to plants, due to their multi-scale nature. Indeed, a plant appears as a discontinuous set of surfaces of various sizes, fuzzily distributed in a volume where multiple occlusions take place. Smallest branches will typically be captured with a very low density of points, which adds fuzziness, and making them likely to be connected by mistake to other parts of the structure. Therefore, standard methods fail for plant acquisition.

The goal of this thesis is to first investigate a new method for acquiring the geometry of existing plants. It is also to provide tools for the quantitative evaluatation of the validity and accuracy of the generated reconstruction.

In the context of this work, a software workflow for reconstructing 3D plant architecture and evaluating it has been developed. These tools make it possible to reconstruct faithfully real plant architectures observed with laser scans. Furthermore, they allow to validate reconstructed models against expert reconstructions.

Document organization

The thesis is organized as follows:

- *Chapter 1* describes the different technology developped for digitizing a shape. A focus is made on laser technology. As these tools produce points sampling the surface of the measured object, a number of methods have been developped to processed this points and regenerate a virtual model of the object. Classical methods of reconstruction are thus presented and discussed in this chapter.

- *Chapter 2* introduces the different plant representations that are used in the literature. A classification of the representation, based on the level of com-

plexity, is used as a guiding canvas for the literature review. In particular, we present the Multiscale Tree Graph, used to represent the structure of a plant at different scales that we used in our reconstruction and evaluation procedure.

- *Chapter 3* presents the different techniques of the literature for generating virtual model of plants. They can be roughly classified as manual measurement techniques, sketch-based, image-based, or point-based.

- *Chapter 4* presents our reconstruction pipeline from laser scanner data. We propose a method locally adaptive to the different level of precision of the data that combines a contraction phase and a local point tracking algorithm to retrieve the skeleton of the shape from the laser data.

- *Chapter 5* describes our approach to quantitatively evaluate reconstructed models. We designed a method to compare reconstructed tree models with expert reconstructions. Structural and geometrical criteria are taken into account. Such method makes it possible to assess the accuracy of our reconstruction algorithm.

Chapter 1

Measuring and representing shape as points

In recent years point-based geometry has gained increasing attention as an alternative surface representation, both for efficient rendering and for flexible geometry processing of highly complex 3D models [Kobbelt & Botsch 2004]. Point surfaces consist of a collection of points in 3D space, each of them representing a small surface area. Similar datasets were acquired in the past by various methods such as range imaging, sonar, and photogrammetry systems. With increasing rate of technology, more cost effective and accurate systems to acquire 3D information have been developed, with for instance laser scanners. In most cases, 3D acquisition devices produce a discrete sample of a object, i.e., a collection of point samples. Depending on the acquisition technique, each point sample also carries a number of attributes, such as color or material properties. Modeling algorithms acting on point-based surfaces are often very efficient, due to the fact that each point sample both stores geometry (e.g., position) as well as appearance information (e.g.,color). Therefore, most modeling operations can be performed by only locally altering the point samples [Adams 2006].

In this chapter, we will show that point acquisition and representation are beneficial when dealing with 3D objects. We present a review of main data acquisition technologies in Section 1.1. In Section 1.2, we present more precisely the technology of laser scanner that can create point cloud of geometric samples on the surface of the object. In Section 1.3, we describe the basic point processing algorithms. Several ways of storing and organizing points data in a computer are discussed in Section 1.4. Finally, we summarize various works in the context of general reconstruction from point sets in Section 1.5.

1.1 Measurement techniques

The first step in reconstructing real-world objects is to carry out data acquisition, that directly affect the accuracy of the model. Data acquisition systems are constrained by physical considerations to acquire data from object surface. There are many different methods for capturing surfaces of objects. These methods can be divided into two types: *tactile contact* type and *non-contact* type (as shown in figure 1.1). In non-contact methods, light, sound or magnetic fields are used while, for contact methods, the surfaces are directly touched by using a mechanical probe.

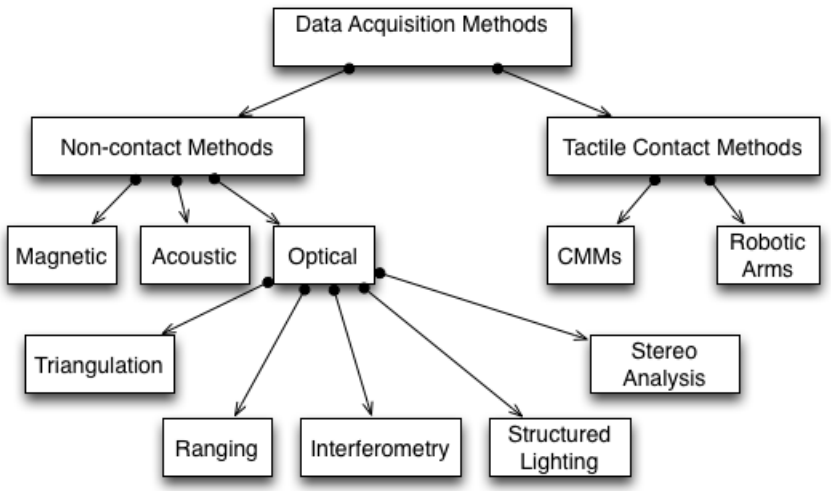


Figure 1.1: Data capturing method. [Várady *et al.* 1997]

1.1.1 Tactile contact methods

The tactile methods, as a contact approach, determined spatial coordinated of object using a mechanical probe to touch a surface and the 3D coordinates of data points on the surface are generated. There are many different robotic devices for inspecting the objects but are not very effective for concave surface. Probably the most popular method is the use of coordinate measuring machines (CMM). These machines provide accurate measuring result and are widely accepted as tools for capturing surface information from objects. Most CMMs are similar to robots equipped with touch probes instead of robot grippers [Spyridi & Requicha 1990]. These machines can be programmed to follow paths along a surface and capture with high accuracy. Several works have been presented using CMMs: Sahoo and Menq [Sahoo & Menq 1991] use CMM system for sensing complex sculptured surfaces, Lin et al. [Lin *et al.* 2005] used this method to reconstruct the CAD model of an artificial join in order to meet their customized demands, etc.

However, CMMs are not suitable for measuring a large number of points since the main limitation of these machines are low speed of data acquisition. They must make physical contact with a part surface for every sampled point. They can also deform a part surface if the part is made of soft material that must not be touched [Wolf *et al.* 2000, Lee *et al.* 2001]. Moreover, these machines are very expensive and complex. Therefore, non-contact methods have been considered for the purpose of data acquistion.

1.1.2 Non-contact methods

Non-contact methods can acquire a large amount of data in a short time compared to a contact device, low cost, moderate accuracy and robust nature in the presence

of ambient light source in situation [Park & Chang 2009]. An early example that is closely related to the use of non-contact methods for measurement is the so-called *Jacob Bar*, see figure 1.2. This measurement is based on the intercept theorem, which involves ratios between segments of a triangle. Such an instrument can either determine the distance from an object (if its lateral dimension is known) or its dimension (if the distance is known) [Schwenke *et al.* 2002].

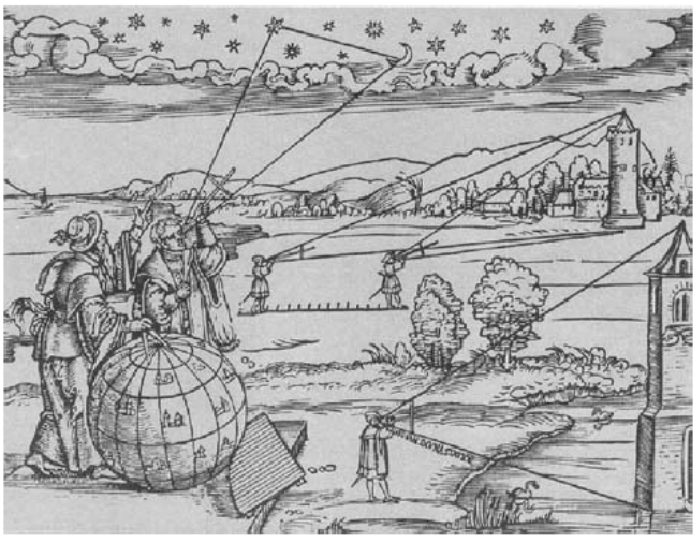


Figure 1.2: The *Jacob Bar* for the determination of distances and lateral dimensions (14th centure) [Schwenke *et al.* 2002].

Later, the rapid growth of computational power, photodiodes and CCD sensor (video cameras) which could convert light intensities into electrical signals have been developed to perform data acquisition. For optical methods, Javis [Jarvis 1983] and Schwenke et al. [Schwenke *et al.* 2002] gave an introduction on the different methods used for data acquisition. It can be summarized as follows.

Five important categories of optical methods are discussed here: triangulation, ranging, interferometry, structured lighting and image analysis.

Triangulation uses the location of an object and angles between light source and detector to deduce position. The light source devices, e.g.,projector, laser scanner, emit light or laser dot on the object and exploit a sensor to look for the position of such dot that appear in the sensor. A point on an object surface can be determined by the trigonometric relations. The position of the point on the detector is a function of the distance between the sensor and the surface of the object. In figure 1.3, let assumes that all geometric parameters are known, the distance from the baseline to the object can be calculated as follow:

$$d = b\frac{\sin\alpha\sin\beta}{\sin(\alpha+\beta)} \tag{1.1}$$

The triangulation method has been applied in various fields. For example, it has

been used to support computer aided quality control processes in manufacturing [Wolf *et al.* 2000]. The main limitation of triangulation is the optical characteristic of the surface, for example very smooth surfaces can not be measured because of insufficient diffuse and reflected light.

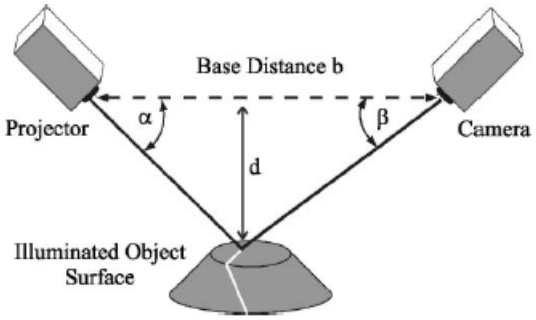


Figure 1.3: Triangulation principle.

Ranging methods are distincted from triangulation methods since the light beams detectors usually sensed returned light and characterized it with the time-of-flight and phase shift of its signals. In this family of method, on can find 3D laser scanners that are used for collecting data on the shape and possible the appearance (i.e.color) of an object. The estimation of the distance to object surface is described in section 1.2.1. This system contributes to achieve high quality data and fast scanning. Moreover, they are capable of operating over long distances. These scanners are thus suitable for measuring large objects. The disadvantage of ranging methods is their accuracy. It can only be used under dry weather conditions: raindrops or fog cause unwanted points and refraction of the laser beam. In section 1.2, more details on 3D laser scanners will be given.

The *Interferometry* [Macgovern & Wyant 1971] method measures distances in term of wavelengths using interference patterns. Most interferometer use light or some other form of electromagnetic wave. This can achieve accurate results since visible light has a wavelenght of the order of hundreds of nanometers, while most optical methods are in the centimeter to meter range [Várady *et al.* 1997]. In principle, a light beam from source will be split into two beams. Each of these beams, a beam to probe the object and a beam to be reference, will travel a different path until they are recombined before arriving at a detector, see figure 1.4. The difference in the distance traveled by each beam creates the interference pattern between the initial waves. A physical change in the path lenght creates a phase difference between the two beams that can be used to estimate the distance.

Nowadays, interferometry is an important investigative technique in the fields of astronomy, fiber optics and biomolecular interactions. However, the device for interferometry are rather complex and expensive. Moreover, dimensions of the investigated object are limited by the size of the objective viewing field and larger deformations lead to formation of a non-distinguishable interference structure.

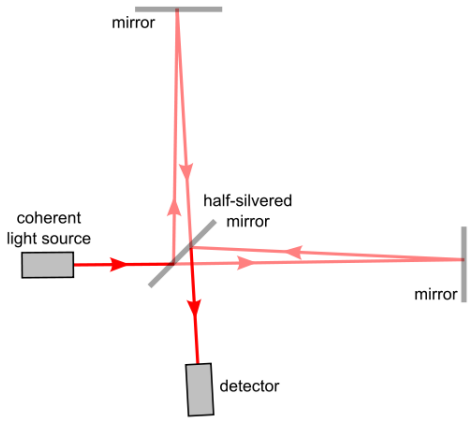


Figure 1.4: An idealized interferometric determination of wavelength obtained by recombining two coherent beams after traveling defferent distance.

Structured lighting involves projected light pattern (often grids or horizontal bars) on to a object and using a camera to capture an image of the resulting pattern as reflected by the surface, see figure 1.5. The light is projected onto the object using either an LCD projector or a sweeping laser. A camera looks at the deformation of the pattern and an algorithm is used to calculate the distance at each point in the pattern, similar to triangulation method. The advantage of this method is its speed. Instead of scanning one point at a time, multiple points or the entire field of view can be scanned at once, but the analysis to determine positions of data can be rather complex. Moreover, much of the work has to be done manually through the intuition of skilled users. Will and Pennington [Will & Pennington 1972] use grid projected onto the surface of objects to determine point locations. Recently, Park and Chang [Park & Chang 2009] proposed a procedure to identify missing areas from multiple scans. This approach attempts to locate additional scanning orientations to fill the missing areas.

Stereo image analysis determines the depth of objects in a scene using two images obtained from two similar cameras that are shifted horizontally [Goshtasby 1989]. If the same object point is available in the two images of the two cameras, then the object point is constrained to lie along two known rays in space and so must lie at their intersection (e.g., see points P1 and P2 in figure 1.6). To determine the three-dimensional positions of points on an object using a pair of images, these positions can be calculated by simple triangulation, as shown in figure 1.6, if the relative positions and orientations of the cameras for the two views are known. The most important and time consuming task of this method is the registration of both images, i.e. the identification of corresponding points.

For *acoustic* measurement system, sonar is used to determine distance between source and objects. The distance is determined knowing the speed of sound, similarly to the estimation of distance from time of flight of light (see equation 1.2). This technique is used to find and identify objects in water. For instance, Giannitrapani

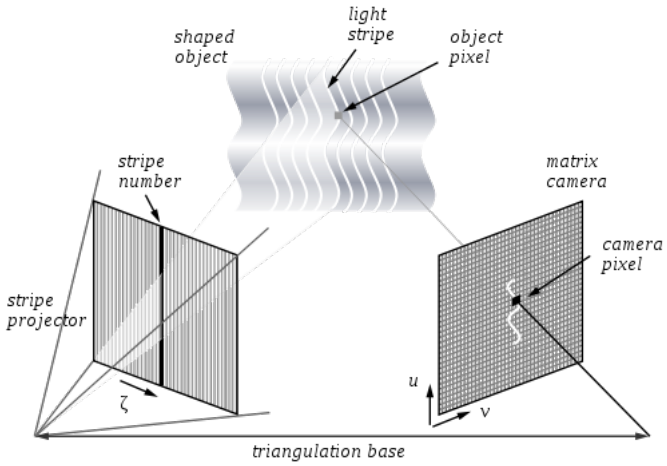


Figure 1.5: Triangulation principle shown by one of multiple stripes.

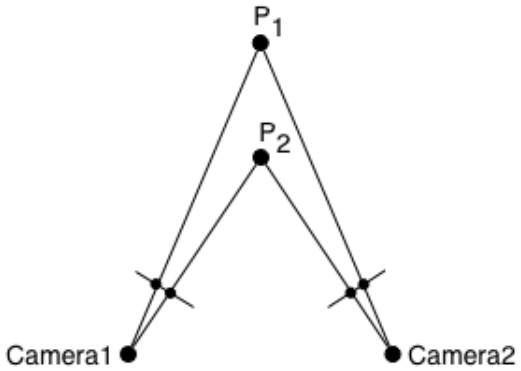


Figure 1.6: Position measurement by stereo triangulation.

et al. [Giannitrapani & Murino 1999] used a high resolution acoustic camera to obtained 3D underwater scene. This technology, making the sound visible, records the sound waves and calculates a sound map for measuring the objects. The main disadvantage of this technology is interference of other sounds.

The last type of data acquisition method is based on *magnetic* field. The magnetic field measurement can be combined with both tactile and non-tactile method. A range measurement device using non-tactile magnetic method is based on triangulation. The coordinates of the measured point are derived from the distance to three fixed points with known coordinates. An alternative tactile method based on electromagnetic field also exists and will be presented in details for the measurement of plant structure in Section 3.1.

All measuring methods must interact with the surface or internal material using some physical phenomenon, either light, sound, magnetism or physical contact. Savio et al. [Savio *et al.* 2007] made a comparision of these measurement techniques. Several criteria such as part dimension, shape complexity, surface conditions and material properties such as hardness and transparency were taken into account to find most appropriate method according to the type of object to measure. The final comparison chart of their study is given in figure 1.7. The most promising method from plant architecture measurement is the laser technology which will be depicted more in details in next section.

1.2 3D Laser scanner

In this thesis, 3D laser scanner were employed for capturing and measuring the geometry of plants. Today, this technology is widespread with numerous fields of application, such as architecture, transportation and engineering. The advantages of laser scanner over traditional techniques, e.g. photogrammetry, are numerous [Reshetyuk 2009]:

- Direct, rapid and detailed capture of object geometry in 3D.
- Possibility to survey remotely very complex, inaccessible and hazardous objects and areas.
- No need for lighting for data acquisition.
- Dramatic reduction in costs and much faster project completion.
- Multipurpose use of the data, both currently and in the future.
- Completeness and comprehensiveness of scanning: everything in the scene is captured at once. Therefore, the user does not need to return to the site if some new data is needed. This also increases the user confidence in the results.

The principle of 3D laser scanner is based on the emission-reception of a laser beam. The emitted laser beam is first deflected by a rotating mirror to scan the

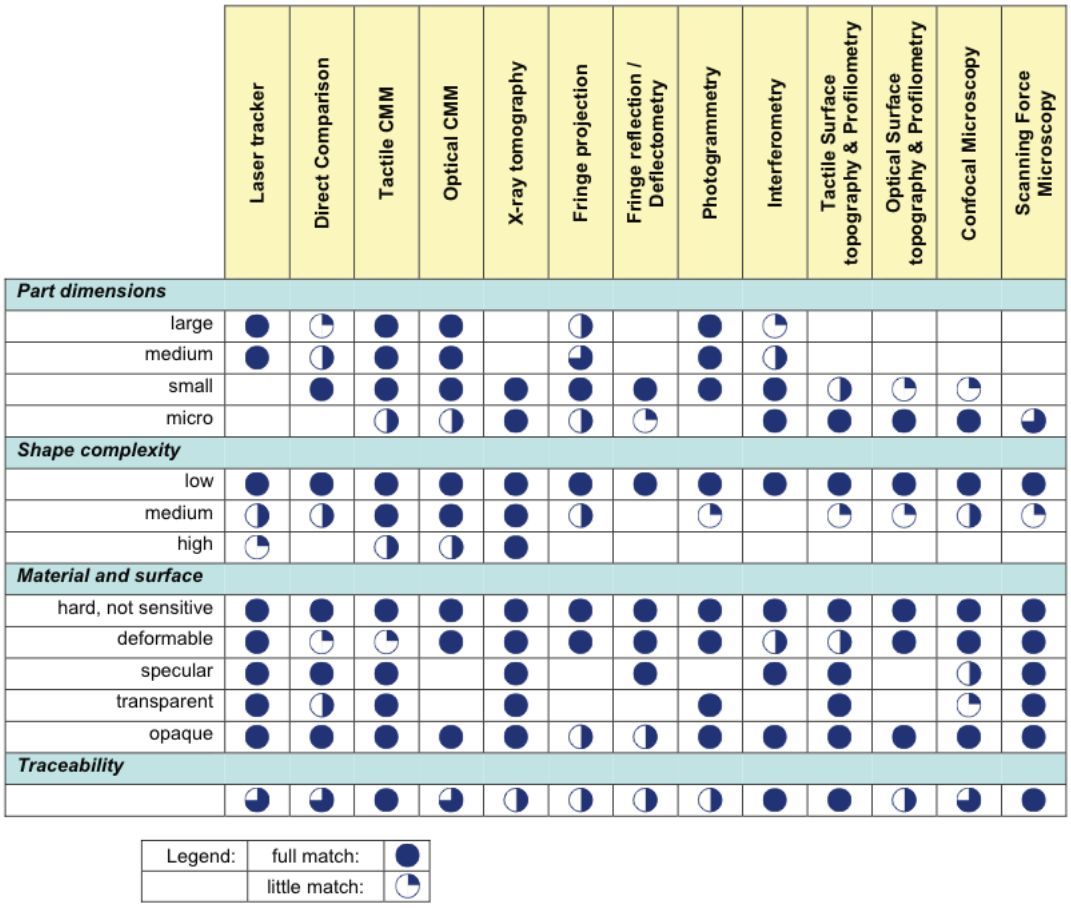


Figure 1.7: Evaluation of some measuring techniques [Savio *et al.* 2007]

surrounding scene in a given direction. The laser beam is then reflected by the first object encountered [Dassot *et al.* 2011] and measured back by a sensor of the scanner. Figure 1.8 illustrates the principle of 3D laser scanner. The laser beam can be automatically deflected by a rotating mirror scans in both horizontal and vertical orientations. Each reflected laser beam result in a record of distance measurement and the creation of a 3D point. The result are a set of 3D points representing the surface of surrounding objects. In next section, we will describe how to determine the distance of an object from the laser beam.

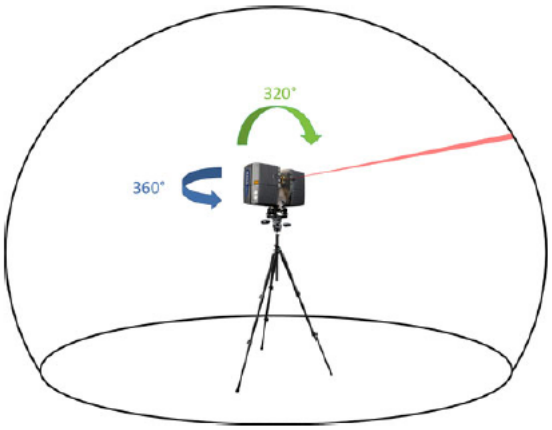


Figure 1.8: Operating principle of 3D laser scanner [Dassot *et al.* 2011]

1.2.1 Range determination

In general, range measurement from laser beam are based on two alternative principle: time of flight or phase shift.

In the time of flight method, laser scanners emit a pulse of laser light that is reflected by the scanned object. The receiver measures the time of flight of travelling light. By knowing the exact value of the light speed, the total time of light from source to the target, see figure 1.9. The distance (D) to the object surface is then calculated as follow:

$$D = (\text{speed of light} * \text{time of flight})/2 \tag{1.2}$$

In the phase shift method, laser beam are emitted with varying wavelength (λ). By comparing emitted and received wavelength of the continuoulsy emitted light beam, a phase shift ($\Delta\phi$) can be determined (see figure 1.10). Let N be the number of full wavelenghts and $\Delta\lambda$ the rest of wavelenght, the range distance can then be computed as $2 * D = N * \lambda + \Delta\lambda$. A relation exists between the rest of wavelenght $\Delta\lambda$ and the rest of phase angle $\Delta\phi$ as $\Delta\lambda = \frac{\Delta\phi}{2\pi} * \lambda$. Therefore, the range distance will be:

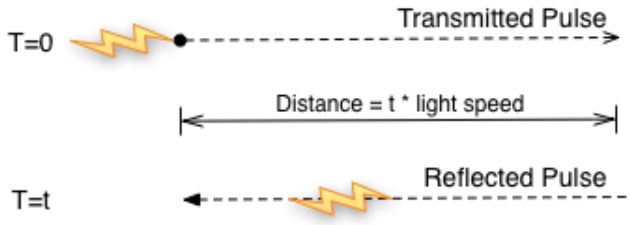


Figure 1.9: Time of flight scanning method.

$$D = \frac{N * \lambda}{2} + \frac{\Delta\phi}{4\pi} * \lambda \tag{1.3}$$

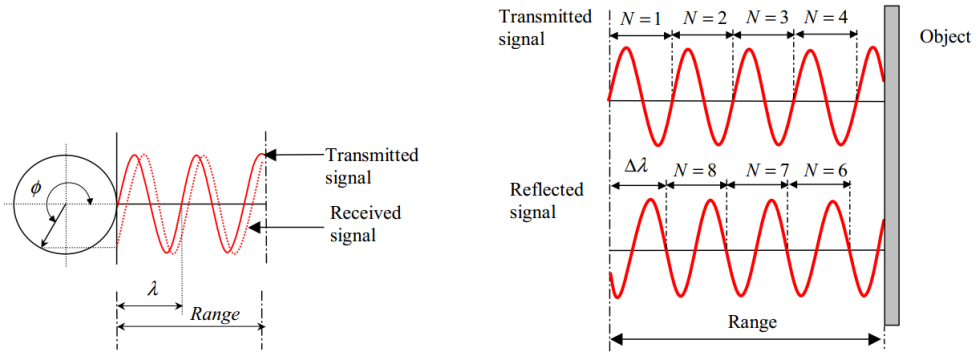


Figure 1.10: Phase comparison measuremant in continuous wave (left). Phase difference techniques measuring (right). [Elkhrachy 2008]

A comparison between these two methods has been carried in[Shahram & Saeed 2006]. The result shows that phase shift technology have faster scanning speed than time of flight but points data obtained from phase shift method contains more noise. The scanners based on the time of flight take usually more time to proceed than scanners based on phase shift.

1.2.2 Coordinate system and transformation

In order to capture the objects in form of point clouds, each laser scanner comes with internal coordinate system. Balis et al. [Balis *et al.* 2004] specify the reference system of a scanning device as follows. The origin is the optical centre of the scanner. The y-axis is the first ray direction. The x-axis is the axis of vertical rotation of the measuring head; the positive direction is selected to make the system right handed. The z-axis is simply perpendicular to the plane defined by the x and y axis, see figure 1.11.

One should note that, due to the differences in the scanner designs, scanners of different manufacturers would have different definitions of the reference coordinate

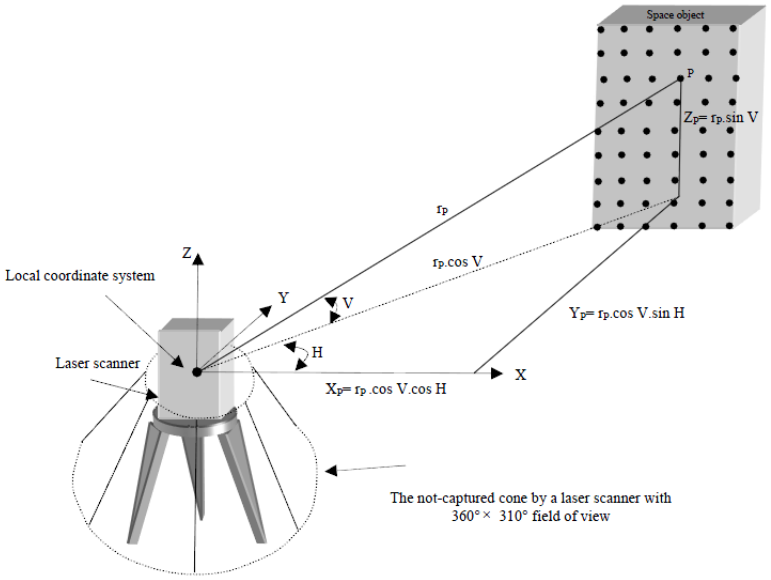


Figure 1.11: Measuring points coordinated by a laser scanner [Abdelhafiz 2009]

system.

In figure 1.11, the laser scanner detects the returned signals of reflections on a surface and records the two directional angles (horizontal angle H, vertical angle V) and the measure range r_p to the object surface. These spherical coordinates fully describe the three dimensional position of each point at the scan object in a local coordinate system relative to the scanner stand point. The transformation to cartesian coordinate system, $[X_p, Y_p, Z_p]$, of the point cloud are thus simply computed as following:

$$\begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} = \begin{bmatrix} r_p \cos V \cos H \\ r_p \cos V \sin H \\ r_p \ sinV \end{bmatrix} \tag{1.4}$$

1.2.3 Points registration

Ususally, the surface of a complex geometry object cannot be scanned completely in just one step, since there are hidden surfaces when viewing in one direction. Therefore, multiple scanning from different point of view is needed to complete the full surface information and a registration step is then requires to fuse all the scans.

More precisely, the *registration* procedure transforms scan data of different scans of the same object onto a common coordinate system [David Barber & Bryan 2001]. Each pixel of the scan has an associated XYZ position expressed in the coordinates system of the scan. Figure 1.12 shows two scans of the object taken from different positions. To determine a common coordinates system, corresponding points in overlap area of different scans are identified. In order to transform the Scan 2 into

the coordinate system of Scan 1, the transformation parameters between these two coordinate system are determined from the different coordinates of corresponding points. The resulting parameters of the transformation consist of a translation along the 3 coordinate axes $(\Delta X, \Delta Y, \Delta Z)$ and rotations around the 3 coordinate axes: (ω, ϕ, κ), as shown in figure 1.12.

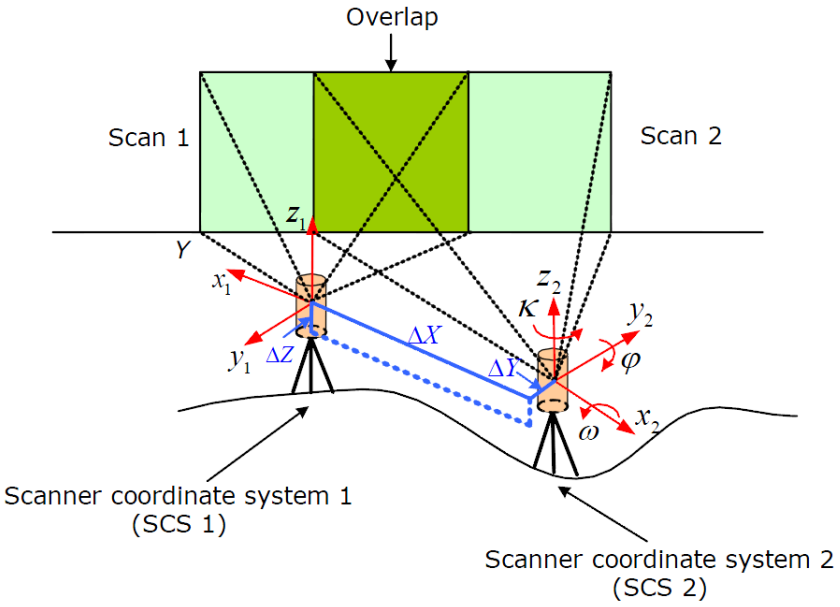


Figure 1.12: Registration of two point sets [Reshetyuk 2009]

Points registration method from multiple scans can be classified into the following three categories: target based, feature based registration [Abdelhafiz 2009].

1.2.3.1 Target based registration

In order to determine the transformation parameters between Scan 1 and Scan 2, this approach needs to know at least 3 points distributed not on the same line, in both Scan 1 and Scan 2. In pratice, the points (T1, T2, T3) with known 3D coordinates are located in the overlap area between two point clouds as figure 1.13. The registration is then computed as a result of the least squares adjustment.

The points can be either natural or artificial targets. Natural targets, for examples edges of steel, corners of buildings and windows, are identified manually, while artificial targets can be assigned automatically by detecting certain shape of targets like spheres or black and white targets. Figure 1.14 shows different types of used artificial targets. The advantage of the use of targets is that they increase the accuracy of the final 3D model. The main disadvantage of this method is the time required for setting the targets before the scanning step. Also, the target should be rescanned with a very fine scanning process to get accurate target parameters. Leica scanning system are able to rescan the reflective targets [Elkhrachy 2008].

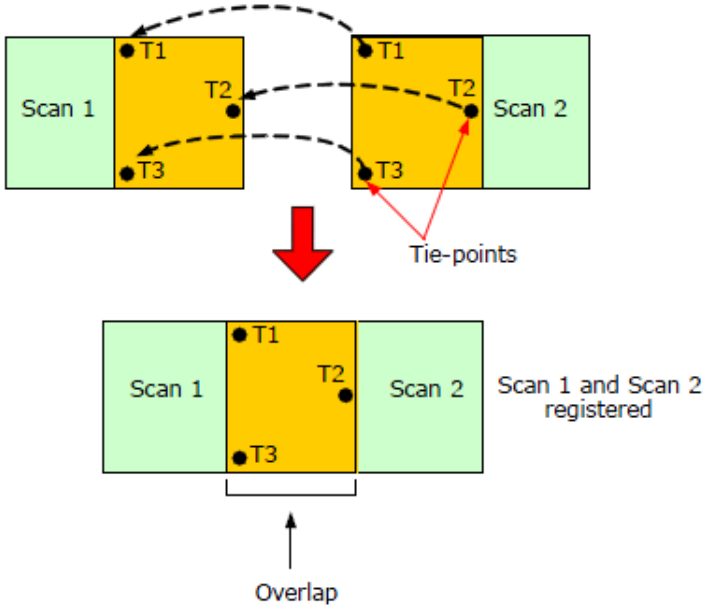


Figure 1.13: Registration using targets [Reshetyuk 2009]

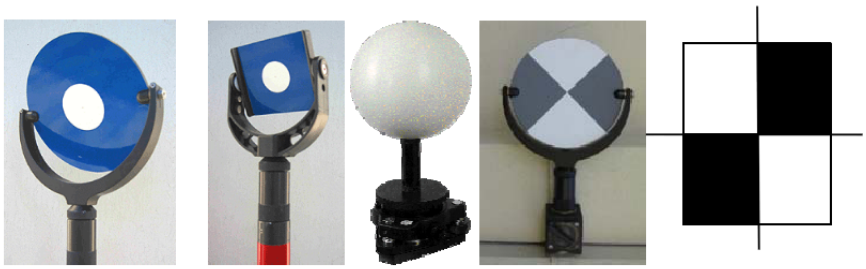


Figure 1.14: Different types of artificial targets [Abdelhafiz 2009]

1.2.3.2 Points based registration

When scanning an object, a 3D laser scanner obtains a huge number of points. So instead of using natural or artificial targets, redundancy points obtained from scanning in overlapping area is considered for points matching. With this data registration technique, all the adjacent scans must have some overlap. Two point sets are iteratively refined by choosing corresponding points, and finding the best transformation based on the distance between the corresponding points.

The Iterative Closest Point (ICP), originally proposed by [Besl & McKay 1992], is a standard algorithm to register laser scanner data. This method performs a recursive minimization of the sum of distances between corresponding points in the different scans. In each iteration step, the algorithm matches the closest points as correspondences and calculates the transformation (translation, rotation) to minimize the distance between two point sets, see figure 1.15. The pointset are then transformed according to the transformation and the process is repeated until convergence.

The advantages of data registration based on point clouds could be summarized as follow:

- No more time and cost as artificial or natural targets are not required.
- No more special scanning processes for targets identification.

While being very simple, ICP method has the following drawbacks:

- It is quite computationally demanding to identify the closest point for each object point at each iteration.
- It does not work properly for some specific shapes (as was mentioned in [Besl & McKay 1992]), though it is a minor and rare problem.
- It can be sensitive to outliers.

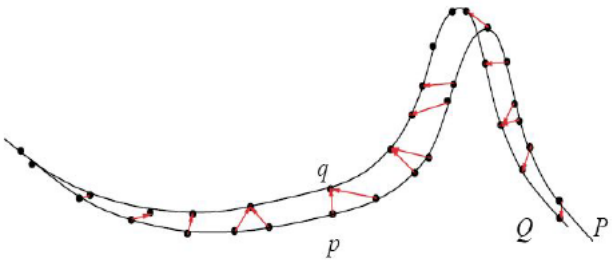


Figure 1.15: One step of the ICP algorithm. Point matches are defined based on shortest euclidean distance. Scan P is then transformed to minimize the length of the displacement vector, in the least-squares sense.

1.3 Points primitives

Over the last decade, point-based data have become popular in computer graphics and related areas. In most case, 3D acquisition create a finite set of points that samples the 3D position of the underlying surface [Andersson *et al.* 2004]. To deal with point-based data for modeling and rendering, we will introduce useful definition describing local neighborhoods and information about normals to the surface that is often not obtained during the acquisition process.

The input data is given as a point set $P = p_i, i \in 1, 2, \ldots, n$. The data points can be produced by some 3D laser scanners or sampling process that creates points from a particular surface. Such data can be associated with color or material properties.

1.3.1 Local neighborhoods

A point-based representation, obtained from devices, has typically no connectivity information available. A neighborhood relation for each point can however be estimated. Given a point $p \in P$, a local neighborhood is defined as an index set N_p such that each $p_i, i \in N_p$ satisfies a certain neighborhood condition. Usually, 3 types of neighborhood are considered: K-nearest neighbors, Delaunay neighbors, and angle criterion neighbors, as shown in figure 1.16.

K-nearest neighbors

A simple method is to form the neighbourhood N_p of a given point p from its k closest points when there is no prior knowledge about the distribution of the data. The definition of the k-nearest neighbors is thus only based on an ordering of all points in P according to the euclidean distance to p. The set N_p^k defines a sphere s_p^k centered at p with radius $r_p^k = \|p_k - p\|$, such that pi is inside s_p^k if and only if $i \in N_p^k$ [Pauly 2003].

The next two methods refine the k-neighborhood method using a tangent plane T of the point. This plane is estimated as the one which minimize the distance to the point of N_p.

Voronoi neighbors

This method attempts to optimize the choice of a neighborhood by using Voronoi diagram.A Voronoi diagram (see [Preparata & Shamos 1985]) of a set of 'sites' (points) is a collection of regions that divide up the plane. Each region corresponds to one of the sites, and all the points in one region are closer to the corresponding site than to any other site. Let q_i be the projection of $p_i \in N_p^k$ onto the tangent plane T of p, and V be the Voronoi diagram of the projected points q_i. The Voronoi cell V_i of q_i is defined as

$$V_i = \{x \in T_p | \|x - q_i\| \leq \|x - q_j\| \forall j \in N_p^k, j \neq i\} \tag{1.5}$$

Let V_p be the Voronoi cell that contains a point p. The Voronoi neighbors of p are defined by the index set $N_p^V \subseteq N_p^k$ such that $i \in N_p^V$, if $i \in N_p^k$ and V_i is adjacent to V_p, i.e., $V_i \cap V_p \neq \varnothing$. [Pauly 2003]

Angle criterion neighbors

Linsen [Linsen 2001] proposed angle criterion method. This approach refine the k-neighborhood of a point as a triangle fan by using an angle criterion. First all points of the k-neighborhood are projected into T. Then the indices of $p_1, \ldots, p_k$ are permuted such that the projections q_i of p_i lead to an increasing sequence of angles $\varphi_i = \angle q_1 q q_i$, where q is the projection of p. If the angle criterion can be satisfied, a new point is inserted into neiborhood of p.

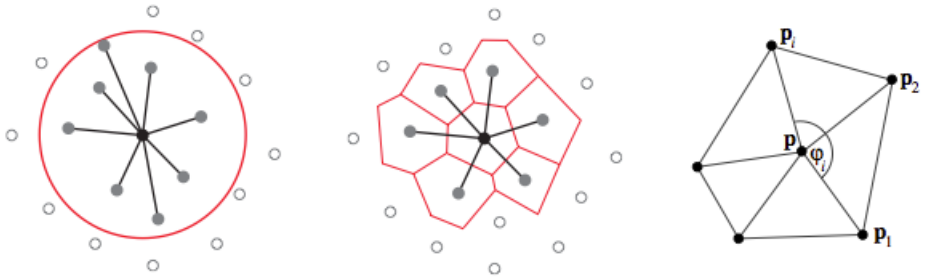


Figure 1.16: Local neighborhoods: (left)k-nearest neighbors according to Euclidean distance, (middle) Voronoi neighbors, (right) Angle criterion neighbors.

1.3.2 Point normals

Given a points p of a surface S, the point normal is a vector perpendicular to the tangent plane to the surface at a point p. The normal vector n_i of a point p can be derived using local covariance analysis of k-nearest neighbors $N_p^k = \{p_1, \ldots, p_k\}$. The covariance matrix is defined as:

$$C = \sum_{j=1}^{k} (p_j - \bar{p})(p_j - \bar{p})^T, \tag{1.6}$$

with $\bar{p} = \frac{1}{k} \sum_{j=1}^{k} p_j$ the average of all neighbor positions. This matrix is a positive semi-definite symmetric 3x3 matrix and therefore all its eigenvalues are real. The eigenvector corresponding to the smallest eigenvalue gives an estimate of the direction of the surface normal for the point p.

A proper orientation of the point normal direction can be computed using a method based on a minimum spanning tree of the point data with distance between point redefined as difference between normal, as described in [Hoppe *et al.* 1992]. The algorithm starts with a point that its normal orients to point away from the centroid of the point cloud. The normal vector of each adjacent point in the minimum spanning tree can then be reoriented based on the assumption that the angle of the normal vectors of adjacent points should be less than $\pi/2$. If the angle to the next adjacent oriented normal is larger than $\pi/2$, the normals is flipped, see figure 1.17. If the point distribution is sufficiently dense, then a consistent orientation

of the normals will be obtained after points of the point cloud have been visited [Pauly 2003].

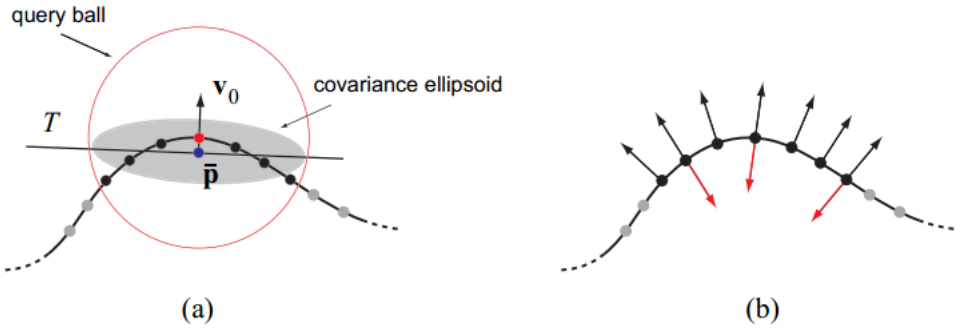


Figure 1.17: Normal estimation (2D for illustration). (a) Computing the tangent plane using covariance analysis, (b) normal orientation using minimum spanning tree, where the red normals have been flipped, since the angle to the next adjacent oriented normal is larger than $\pi/2$. [Pauly 2003]

1.4 Point data structure

The requirements for a data structures to manage the point set considerably depend on the context of use. For instance to determine neighborhoods of all points, an efficient data structure is required. In this section, several classical data structure for organizing and manipulating 3D points set are presented.

1.4.1 Grid based

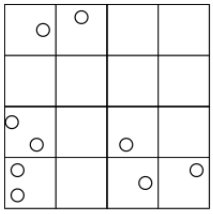


Figure 1.18: grid

The most simple data structure to store point data is a grid (see Figure 2.8) [Samet 1990]. It divides space in equally sized cells, (also called buckets). Each cell can store multiple points and can be implemented as a linked list. The main advantage of grid based data structure is points insertion and deletion that remains straightforward. However, it loses its interest if points spatial distribution is non-uniform. If the buckets are unevenly distributed, an important memory space may be wasted. Moreover, point queries can become inefficient since the data stored in

the grid are partitioned independently and may consume an unpredictable amount of time. Finally, defining the grid might be too memory consuming.

1.4.2 Octree

Similar to the grid, the octree is a space partitioning data structure. The main difference is that the partitioning is built in a top-down approach by recursive subdivision of the space in eight subcubes (octants). In case of empty regions, the subdivision is stopped. Else the voxels are subdivided until reaching a minimal number of points.

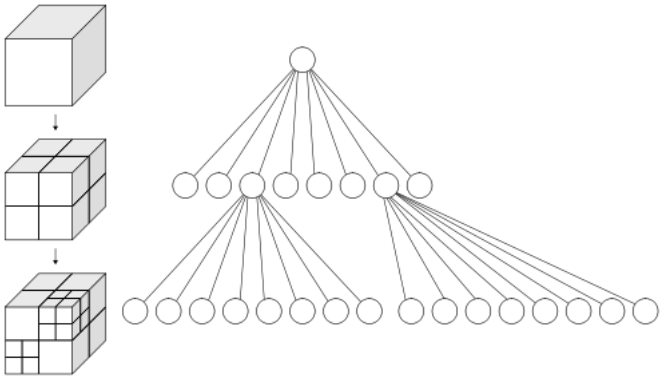


Figure 1.19: Octree

The octree has advantage over the grid for quering nearest neighbor. However, they have the major disadvantage of constructing the tree based on 3-D points data, It is not a balanced tree since the octree can have different of depths according to distribution of data. In case of query, the three coordinates of the query point must to be tested at each level in order to determine the exact suboctant when traversing in the tree.

1.4.3 K-d tree

The k-d tree [Bentley 1975, Friedman *et al.* 1977, Arya & Mount 1993] is similar to the octree presented in the previous except that it is a binary tree. The space is recursively subdivided using hyperplanes that are perpendicular to the coordinate axes. For example, the first plane is perpendicular to the X axis, the second plane is perpendicular to the Y axis and the third to the Z axis.

The k-d tree is constructed as follows: Initially, an empty root node which represents the bounding box of the whole scene is created. At the root, the point set is divided into two equally sized subsets according to a plane perpendicular to the first axis, i.e., X axis. Next, two new splits are made independently according to the second axis in each part respectivelly. The next axis in then used to split again independently the different subsets.

The splitting process is done recursivelly and terminated when the graph depth is higher than or equal to a fixed constant, or every splitted box contained only a

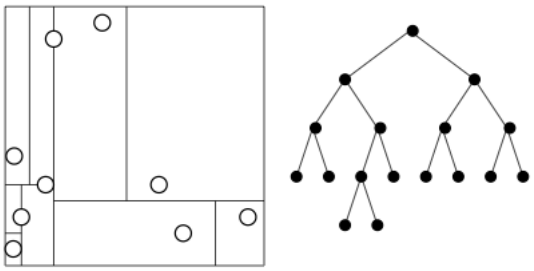


Figure 1.20: K-d tree

certain number of objects.

Table 1.1 compares the average time complexity of these data structure for typical operations used on points data.

Table 1.1: Averate time complexity for spatial data structures used for closest point queries. n is the number of points in P, m is the average number of points in each grid cell. Update refer to a change of sample position or a sample point deletion [Pauly 2003].

Data structure	Construction	Insertion	Update	Query
List	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Grid	$O(n)$	$O(1)$	$O(1)$	$O(m)$
K-d tree	$O(n \log n)$	$O(\log n)$	$O(\log n)$	$O(\log n)$
Octree	$O(n \log n)$	$O(\log n)$	$O(\log n)$	$O(\log n)$

1.5 General reconstruction methods from point sets

A numbers of methods for shape reconstruction from point sets have been proposed. These methods try to reconstruct either the surface or the skeleton of the shape. These two types of methods will be presented in the next sections.

1.5.1 Surfaces reconstruction

Point sets have typically emerged as a surface representation. Nevertheless, most surfaces have many features or forms which restricts the possibilities of using directly functions for interpolation or approximation. Alternative methods have been thus proposed.

One of the most popular technique to define a manifold surface from a set of points is the *Point Set Surface* (PSS) model. This technique was initially proposed by Alexa et al. [Alexa *et al.* 2001, Alexa *et al.* 2003]. Based on the *Moving Least Squares* (MLS) [Levin 1998, Levin 2003], this technique defines a smooth surface using projection procedure on point set. The main idea is that the procedure projects any points in the neighborhood of the point set onto the surface, that minimizes a

local least squares error criterion. The MLS projection has two step procedure, (see figure 1.21). First, a local reference domain H for the purple point r is generated. The projection of r onto H defines its origin q (the red point). Then, a local polynomial approximation g to the heights f_i of points p_i is a function of the distance to q. The projection of r onto g (the blue point) is the result of the MLS projection procedure.

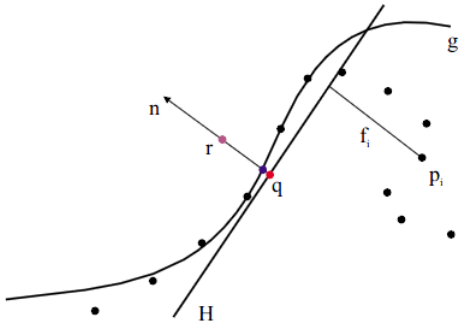


Figure 1.21: The MLS projection procedure. [Alexa *et al.* 2001]

PSS model are not only used to define a smooth surface from a point set, but they are also a tool to decimate or generate additional sample points on a surface. The decimation process removes the point that contributes the smallest amount of information to the shape[Alexa *et al.* 2001]. In the case of undersampling, the input point set needs to be up-sampled. The basic idea of additional points process is to compute Voronoi diagrams on the MLS surface and add points at vertices of this diagram. This idea is related to Lloyd's method [Lloyd 1982], i.e., techniques using Voronoi diagrams to achieve a certain distribution of points [Okabe *et al.* 2000]. Figure 1.22 illustrates the original point set and the same object after adding points over its MLS surface.

The drawback of MLS surfaces are two fold: i) the reference plane fit operation is highly unstable in regions of high curvature if the points density drops below a threshold. ii) a projection operation based on only the first step is not orthogonal since the surface normal of the approximating plane is not true.

As limitation of the robustness of PSS, a slightly different but considerably simpler projection approach is proposed by Alexa and Adamson [Alexa & Adamson 2004], where they also take a correct normal computation into account in order to properly define implicit surfaces from point cloud data. Also, Guennebaud and Gross [Guennebaud & Gross 2007] proposed an algebraic point set surfaces (APSS) framework to locally approximate the data using algebraic spheres. The key idea is to directly fit a higher order algebraic surface [Pratt 1987] rather than a plane. This work improve stability of the projection under low sampling densities and in the presence of high curvature.

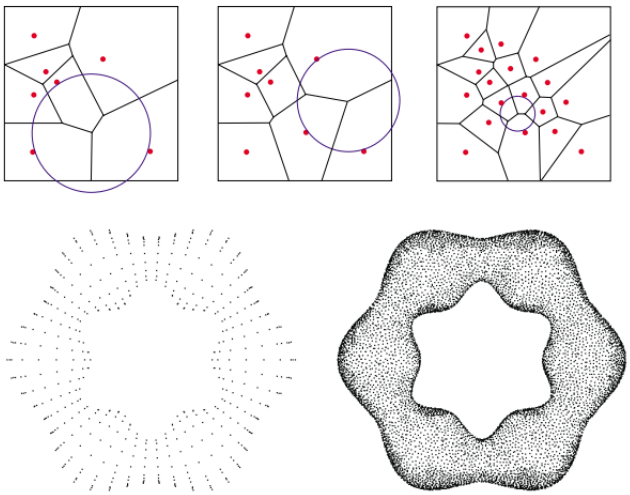


Figure 1.22: The up-sampling process: Points are added at vertices of the Voronoi diagram. In each step, the vertex with the largest empty circle is chosen. The process is repeated until the radius of the largest circle is smaller than a specified bound. The original object consisting of 800 points has been up-sampled to 20K points. [Alexa *et al.* 2001]

1.5.2 Skeleton reconstruction

1.5.2.1 Skeleton definition

Curve skeletons are thinned 1D structures that represent a simplified version of the geometry and topology of a 3D objects. The skeleton is defined as the locus of centers of maximal inscribed (open) balls (or disks in 2D) [Lieutier 2003, Cornea *et al.* 2007]. More formally, Cornea et al. [Cornea *et al.* 2007] have given a definition of the skeleton as follow:

Let $X \subset R^3$ be a 3D shape. An (open) ball of radius r centered at $x \in X$ is defined as $Sr(x) = y \in R^3, d(x, y) < r$, where $d(x, y)$ is the distance between two points x and y in R^3. A ball $Sr(x) \subset X$ is maximal if it is not completely included in any other ball included in X [Kong & Rosenfeld 1989]. The skeleton is then the set of centers of all maximal balls included in X. The process of obtaining a skeleton is called *skeletonization*. Figure 1.23 shows curve skeletons of several 3D objects

Medial axis is closely related to skeleton. The medial axis of a closed surface is the set of centers of empty balls which touch the surface at more than one point. On the ellipse of figure 1.24, for example given in [Lieutier 2003], both the medial axis and the skeleton of a 2D ellipse are represented by a line segment but the end points of the segment belong to the skeleton, while they do not belong to the medial axis.

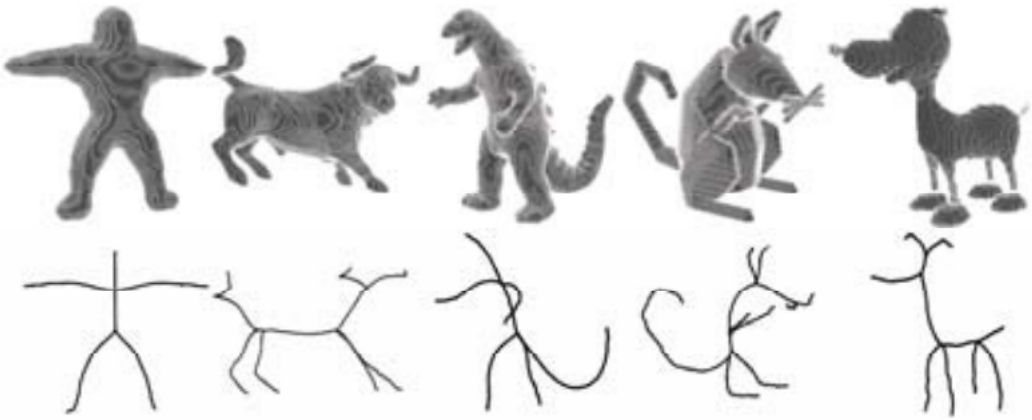


Figure 1.23: Examples of curve skeletons of different 3D objects. [Cornea *et al.* 2007]

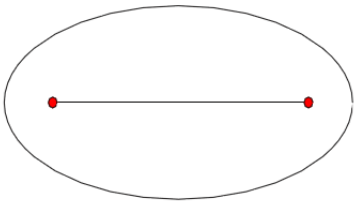


Figure 1.24: The end points of the segment belong to the skeleton, not to the medial axis.

1.5.2.2 Skeleton reconstruction based on grid

A first set of methods to produce skeleton are based on objects represented by voxels on a regular grid. These methods can be classified in the following way:

Topological thinning method: This method attempt to produce a curve skeleton by iteratively removing surface voxels from the boundary of an object without changing the topology of the object until the required thinness is obtained [Cornea *et al.* 2007]. In each iteration, to prevent over thinning of the object, deletion of surface voxel must satisfy topology preserving conditions. The algorithm is stopped when no more voxels can be removed. Figure 1.25 shows an object like a tree-way cross and its skeleton produced from thinning method.

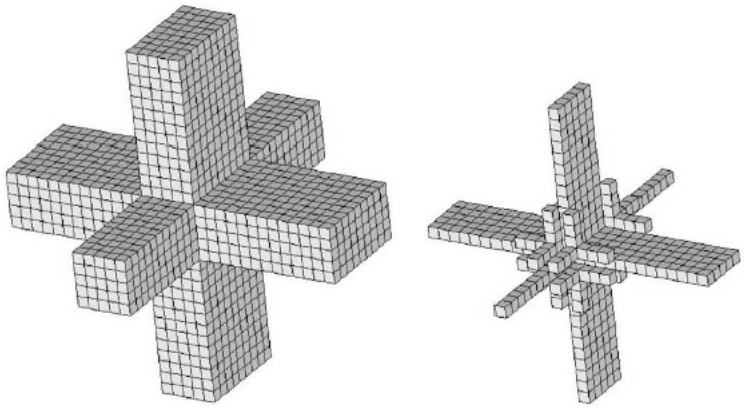


Figure 1.25: A 3D object and its skeleton. [Ma & Wan 2001]

In general, thinning methods produce connected curve skeletons that preserve the object topology. However, it is possible to produce excessive shortening of the curve skeleton branches. In addition, preserving more voxels leads to many undesired small branches.

Distance transform method: These methods compute the boundary's distance transform (DT) and define the skeleton as the DT's local maxima, or 'ridges' [Telea & Vilanova 2003]. Several DT can be used such as the Euclidean distance, or an approximation using chamfer metric (3,4,5) [Borgefors 1986], or an approximation using the fast marching methods [Adalsteinsson & Sethian 1995]. Most of the distance field-based algorithms have these three steps: (1) generate the distance field and find the ridges (candidate voxels) of this distance field, (2) prune the set of candidate voxels down to a manageable size, (3) connect the remaining voxels to produce curve skeletons. Figure 3.5 shows the color-coded distance field values on a slice of a 3D chess piece shape. The color map ranges from blue for small distance field values to red for large values. The ridge voxels of the distance transform are locally centered with respect to the object's boundary.

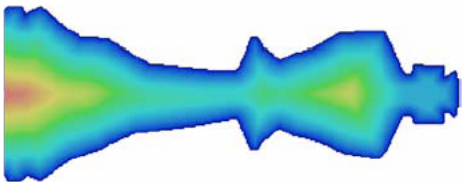


Figure 1.26: A color-coded slice of the distance field of a 3D shape. [Cornea *et al.* 2007]

1.5.2.3 Skeleton reconstruction based on mesh

Object can be represented by their boundary surface generally described by polygonal meshes as shown in figure 1.27. Various methods were proposed to create skeleton from input mesh. These works can be organized into the following four categories:

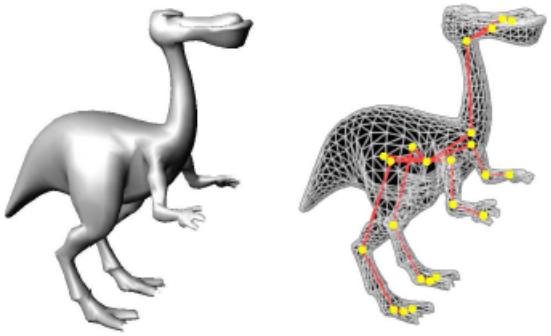


Figure 1.27: An example of extracting skeleton based on mesh: (left) An object, (right) skeleton. [Katz & Tal 2003]

Surface/volume decomposition-based methods: Katz and Tal [Katz & Tal 2003] extract a skeleton from a meshed model using a hierarchical mesh decomposition algorithm. The given object is decomposed into simpler sub-objects using an iterative clustering scheme. The skeleton extraction is then computed from these components. Lien et al. [Lien *et al.* 2006] presented an iterative approach that simultaneously generates a shape decomposition and a corresponding set of skeletons. The key ideas of the algorithm have three steps. First, a simple skeleton is extracted from the decomposition components of object using the principal axis of the convex hull of component. Then, this extracted skeleton is used to evaluate the quality of the decomposition. Finally, if the skeleton is satisfactory under some user defined criteria, the skeleton and the decomposition are reported as final results.

Force field-based methods: Liu et al. [Liu *et al.* 2003] proposed to use repulsive force field to automatically generate the animation skeleton of a model. Initially, a modified voxelization is used to construct the repulsive force field, with the resolution being defined by voxel-size parameter. The repulsive force field is computed by

hundreds of rays that are shot radially from sample points inside the model. After obtaining the force field, the force magnitude of each point is compared with the neighbors. The points with local minimal force magnitude are chosen as the skeleton joint candidates. Then, a modified thinning algorithm is applied to generate an refined skeleton. Wu et al. [Wu *et al.* 2006] combines a medial axis approach with a decomposition and potential field for creating a concise structure to represent the control skeleton of an arbitrary object.

Reeb graph-based methods: Reeb [Reeb 1946] has defined a skeleton structure, called the Reeb graph. The idea of Reeb graph is to use a continuous function, usually a height function to describe the topological structure and reveal the topological changes such as splitting and merging. The result of a Reeb graph highly depends on the choice of these functions. Based on Reeb graph, [Yang *et al.* 2005] proposed an approach to extract skeletons of tubular structures and applied it to both healthy and diseased coronary arteries. Moreover, [Aujay *et al.* 2007] presented a fully automatic method to compute an animation skeleton from a 3D meshed model.

Example-based methods: These method construct a skeleton for a given shape based on already created skeletons for the same shape under different poses. The first approach has been raised by Schaefer and Yuksel [Schaefer & Yuksel 2007]. They present a method for extracting a hierarchical, rigid skeleton from a set of example poses. They use this skeleton to not only reproduce the example poses, but create new deformations in the same style as the examples, see figure 1.28. Then, He et al. [He *et al.* 2009] proposed a method to easily find a consistent mapping between the reference and example poses. Recently, Hasler et al. [Hasler *et al.* 2010] were able to combine the information from example sets from different subjects to improve the stability of the method.

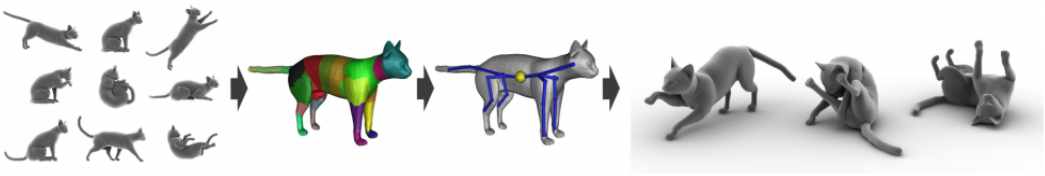


Figure 1.28: From left to right: Example poses of a cat, clustering for bone transformations, skeleton found with root shown in yellow, and new poses created using this skeleton. [Schaefer & Yuksel 2007]

1.5.2.4 Skeleton reconstruction based on unorganized points

Surface of objects can also be represented as cloud of points. In general, point-based representations lack information on the structure of the object which can be retrieved with the skeleton. Several methods for extracting skeleton from unorganized points have been proposed.

An algorithm to extract a skeleton on-the-fly, both from point clouds and polygonal meshes has been presented by Sharf et al. [Sharf *et al.* 2007]. The key idea is

to track the reconstruction of a given object using a deformable model with competing evolving fronts [Sharf *et al.* 2006]. The front evolution is guided by a scalar-field representing the distance from the point set. The evolution process starts by placing a sphere-like mesh inside the shape. The algorithm proceeds by iteratively advancing the initial mesh towards the shape. At each iteration a new skeleton node for every front at the barycenter of its mesh vertices is inserted. Nevertheless, the curve-skeleton can become very noisy and contain spurious branching. They allow to simplify the initial skeleton connectivity by pruning and merging. Using user specified parameter, the skeleton can also be filtered to obtain different level-of-details.

Figure 1.29: Computation of cross-section plane for each sample point. Starting from an initial cutting plane (left), the algorithm iteratively re-estimate the plane (right) using a local optimization. [Sharf *et al.* 2007]

Tagliasacchi et al.[Tagliasacchi *et al.* 2009] develop an algorithm for curve skeleton extraction from incomplete point clouds. They assume that the input point cloud samples a shape which is composed of generally cylindrical regions, called branch regions, except at joint regions, as shown in figure 1.30(a). A rotational symmetry axis (direction) on a branch region can be represented in term of a cutting plane, perpendicular to the symmetry axis. Intuitively, a rotational symmetry center should lie on this plane, see figure 1.30(b). For example, given the samples lying on a candidate cutting plane: (1) the axis minimizes the sum of angular distances with the sample normals. (2) The center minimizes the sum of distances to normal lines, see figure 1.30(c).

Motivated by the work of [Tagliasacchi *et al.* 2009], a deformable model, called arterial snake, was proposed by Li et al. [Li *et al.* 2010]. Given an input scanning data, a cutting plane for each point is iteratively computed using a local optimization and assign the normal of the cutting plane as the vector field over the data point. The orientation vectors in tubular sections are consistent and well-behaved, while there are unstable and noisy near joint regions. The unreliable directional vectors are thus filtered out. Points with reliable directional vectors are grouped into clusters and arterial snake model are fitted into such cluster (see 1.31-leftmost). Snakes are

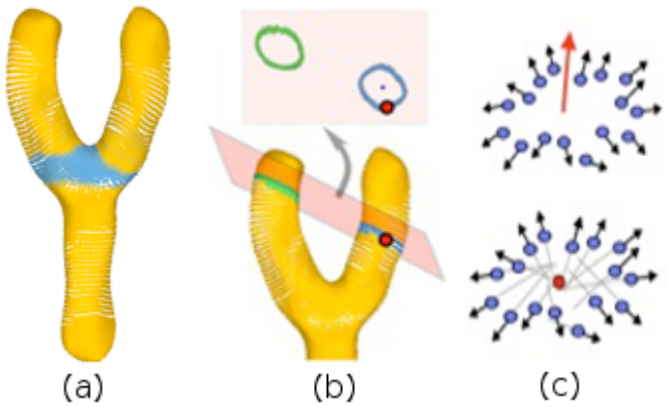


Figure 1.30: (a) Input point cloud with a joint regions. (b) Rotational symmetry for a surface sample. (c) Optimal direction and point on a cutting plane. [Tagliasacchi *et al.* 2009]

then allows to grow. When all snakes stop evolving, the potential snake pairs are merged. They connect two skakes if their end points are close and the respective tangents to the skeletal curves near their end points agree (see 1.31).

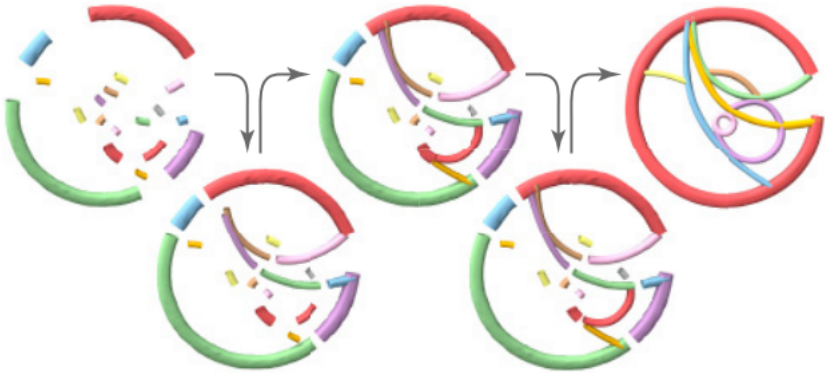


Figure 1.31: Topology recovering of curve network. Starting from reliable snakelets, the arterial snakes grow simultaneously while competing with each other. Longer and flatter curves grow faster. Snakes may retract to make way for others to grow first, and adjacent snakes with suitable tangent conditions merge together.[Li *et al.* 2010]

Cao et al. [Cao *et al.* 2010] present an algorithm for curve skeleton extraction via topological thinning to reduce the skeleton structure to a line, see figure 1.32(b) and (c). This work is based on Au et al. [Au *et al.* 2008] but the method performs directly on point cloud data instead of the mesh domain. The algorithm first contracts a point cloud to a zero-volume point set. The contraction maintains the global shape of the input model by anchoring points chosen by an implicit Laplacian

smoothing process. Correct parameters tuning will result in a contraction which well approximates the original geometry of the shape. Next, a skeleton graph is built by sub-sampling the contracted cloud and computing a restricted connectivity. The skeleton graph with uniformly distributed nodes show in figure 1.32(c). Finally, unnecessary edges on the graph, measured by minimum Euclidean length, are iteratively collapsed until no triangles exist to build a curve skeleton. Figure 1.32 give the overview of this algorithm.

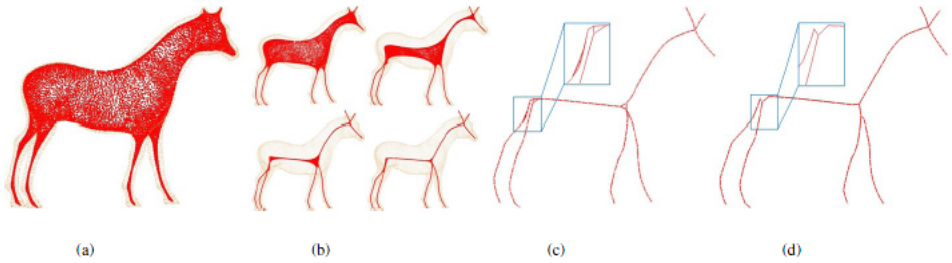


Figure 1.32: Overview of the algorithm. (a) Input point cloud(orange) and the contracted points (red) after 1 iteration. (b) Contracted points after 2-5 iterations. (c) Skeleton graph constructed by father-point sampling (light red points) with a fixed radius ball and connectivity (red lines) inherited from the local neighborhood relationship of the input points. (d) The final 1D curve skeleton after topology thinning.

Conclusion

This chapter introduced how to measure and represent a shape with points. This will be the basis of our work. In particular, the different categories of measurement technologies have been depicted. Our work will focus on laser scanners. First methods on the processing of point sets were also described for instance to manipulate or reconstruct the shape or the skeleton of the measured object.

Chapter 2

Plant architecture representation

In general, representation of *plant architecture* are used to model structure and functionning of plants. However, the meaning of the concept of plant architecture is variable according to the context or the application considered. For instance, Hallé et al. [Hallé *et al.* 1978] consider it as the architectural model of a tree species, i.e. a set of rules that describes the growth patterns of an average individual of species. For Ross [Ross 1981], plant architecture is a "set of features delineating the shape, size, geometry and external structure of a plant". Godin [Godin 2000] proposes a more general definition where plant architecture is any individual description based on decomposition of the plant into components, specifying their biological type and/or their shape, and/or their location/orientation in space and/or the way these components are physically related one with another. This last definition better suits the work presented in this thesis, where 3D structure of real plants are reconstructed. From a reconstruction, the goal is to produce a representation of the architecture of a plant as the shapes and spatial positions of the components of the plant, and the connectivity between these components.

The aim of this chapter is to provide guiding principles of the numerous plant architecture representations. In Section 2.1, we will introduced *global representations* where plants are represented by a unique and compact model. These representations consider the complexity of a plant at the lowest level. By contrast, *detailed representations*, described in Section 2.2, rely on specific decomposition of a plant into modules. These representations have been organized in this review into three classes: spatial representations, topological representations, and geometric representations.

2.1 Global representations

Due to the geometric complexity of plants, modules or organs of similar types of them can be reduced into one or a few compartments providing a simple and compact representation. Therefore, this first approach consists of representing the plant as a whole, not decomposed into modules. These global representations can be divided into two categories: envelope-based and compartment-based representations.

2.1.1 Envelope-based representation

The complexity of plant geometry can often be abstracted as global models representating their crown. The crown shape of a whole plant is a simple representation and has been used in various contexts for studying plant interaction with its environment

(i.e. competition for space [Phattaralerphong & Sinoquet 2005] or radiative transfer in canopies [Cescatti 1997]), 3D tree model reconstruction from photographs [Shlyakhter *et al.* 2001], and interactive rendering [Reche-Martinez *et al.* 2004]. These envelope models were specifically designed to represent plant volumes.

2.1.1.1 Simple envelope

A simple geometrical description of an enveloping surface of the crown shape can be constructed using symmetrical solids such as cones, ellipsoids, cylinders, or a combination of these. Spheres or ellipses are used for instance to model light interception by tree crowns [Norman & Welles 1983], see figure 2.1. The plant canopy is approximated by an array of ellipsoidal subcanopies that may be equally spaced, randomly spaced, or spaced in any desired manner. In Baker's work [Baker 1995], cylinders, cone frustums or paraboloids are used to study the mechanical properties of plants.

The main limitation of such approach are the approximate description of crown asymmetry.

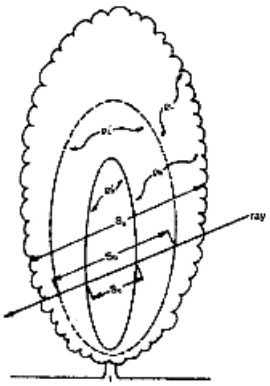


Figure 2.1: Simple envelope representation by a set of ellipses. This model is used to calculate the light interception of the tree

2.1.1.2 Asymmetric envelope

Asymmetric envelope, originally proposed by Horn [Horn 1971] and Koop [Koop 1989], then extended by Cescatti [Cescatti 1997] and Pradal et al. [Pradal *et al.* 2009], makes it possible to easily define asymmmetric crown shapes. These envelope models can adapt to various tree geometries based on the following parameters: total tree height, height at crown inserting and at the greatest width of the crown, crown radii in four orthogonal directions, and shape coefficients of vertical crown profiles.

In figure 2.2, this envelope model can be created using six control points in six directions and two shape factors C_T and C_B that control its convexity. To

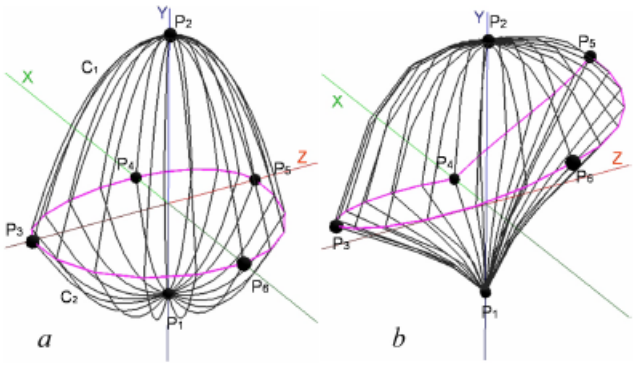


Figure 2.2: Asymmetric hull parameters. [Pradal *et al.* 2009]

define height of the crown, the control point P_T represents the crown apex and P_B represents the crown base. The radius of the crown is defined using the peripheral points P_1 to P_4. P_1 and P_3 are constrained to lie in the xz plane and P_2 and P_4 in the yz plane. Then, the peripheral line L at the greatest width of the crown can be defined from such points.

A point p of L can be computed as

$$p = [r_{P_i} cos\theta, r_{P_j} sin\theta, z_{P_i} cos^2\theta + z_{P_j} sin^2\theta]. \tag{2.1}$$

Where P_i and P_j are consecutive points with $i, j \in [(1,2),(2,3),(3,4),(4,1)]$, and $\theta \in [0, \frac{\pi}{2})$ is an angle between P_i and P_j, and $r_{P_x} = \sqrt{x_{P_x}^2 + y_{P_x}^2}$.

Points of L are connected to the P_T and P_B. The curvature of the crown above and below of the peripheral line L are described with quarters of super-ellipses of degrees C_T and C_B respectively. The super-ellipses quarters connected between a point p of L and P_T is formed as

$$\frac{(r - r_{P_T})^{C_T}}{(r_p - r_{P_T})^{C_T}} + \frac{(z - z_{P_T})^{C_T}}{(z_p - z_{P_T})^{C_T}} = 1 \tag{2.2}$$

An equivalent equation is obtained for super-ellipse quarters using P_B and C_B instead of P_T and C_T.

The shape factors generate different envelope shapes such as conical ($C_i = 1$), ellipsoidal ($C_i = 2$), concave ($C_i \in]0,1[$), or convex ($C_i \in]1,\infty[$) shapes. Figure 2.3 shows the various tree geometries represented using asymmetric envelopes.

2.1.1.3 Extruded envelope

Birnbaum [Birnbaum 1997] proposed extruded envelope from an horizontal and a vertical profiles for plant modelling. In this approach, each crown is represented in two dimensions: the first one is obtained from the projection of tree crown on the ground (see figure 2.4.A), and the second one is obtained from a lateral view of the tree. The vertical profile is virtually split into slices by horizontal planes (see

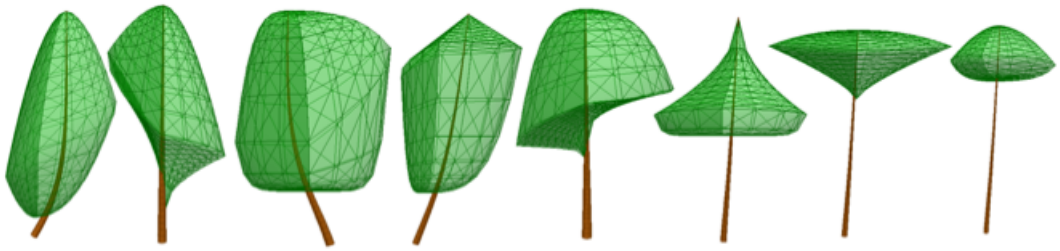


Figure 2.3: Examples of asymmetric hulls showing plasticity to represent tree crowns generated by PlangGL [Pradal *et al.* 2009].

figure 2.4.B) or by planes passing through equidistant points from the top (see figure 2.4.C). These slices are used to map horizontal profile inside vertical profile and a mesh is calculated.

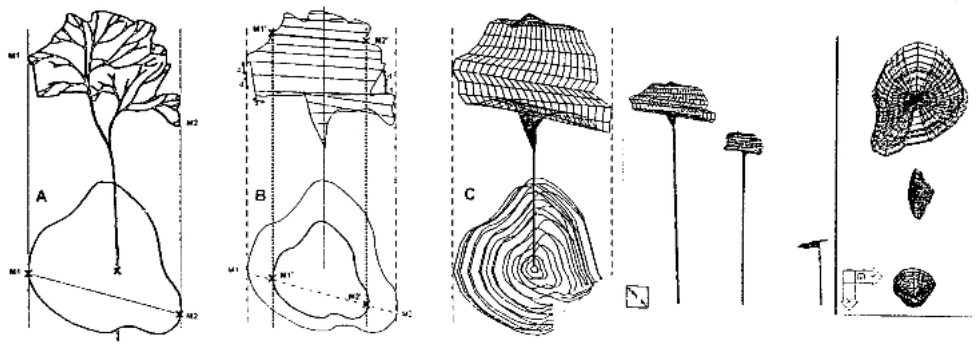


Figure 2.4: Extruded hull envelope contructed from a horizontal and vertical profile. [Birnbaum 1997]

In figure 2.5, generated from PlantGL [Pradal *et al.* 2009], illustrates the reconstruction of extruded envelope by sweeping the horizontal profile H inside the vertical profile V. Given B and T are the fixed points at the bottom and top of the vertical profile respectively. V_l and V_r are open profile curves, left and right, with their first and last points equal to B and T. A slice $S(u)$ is thus defined using two mapping points $V_l(u)$ and $V_r(u)$. Its span vector $\overrightarrow{S(u)}$ is set to $\overrightarrow{V_l(u)V_r(u)}$.

On the horizontal profile H, H_l and H_r, the left and right points, are defined. A horizontal section of the extruded hull is computed at $S(u)$ so that the resulting curve fits inside V (see figure 2.5.b and c). Finally, to map $\overrightarrow{H_lH_r}$ to $\overrightarrow{S(u)}$, the transformation is a composition of a translation from H_l to $V_l(u)$, a rotation around the $\overrightarrow{y}$ axis of an angle $\alpha(u)$ equal to the angle between the $\overrightarrow{x}$ axis and $\overrightarrow{S(u)}$, and a scaling by the factor $\|\overrightarrow{S(u)}\|/\|\overrightarrow{H_lH_r}\|$.

The equation of the extruded hull surface is thus defined in [Pradal *et al.* 2009] as :

$$S(u,v) = V_l(u) + \frac{\|\overrightarrow{S(u)}\|}{|\overrightarrow{H_l H_r}\|} * R_y(\alpha(u))(H(v) - H_l) \tag{2.3}$$

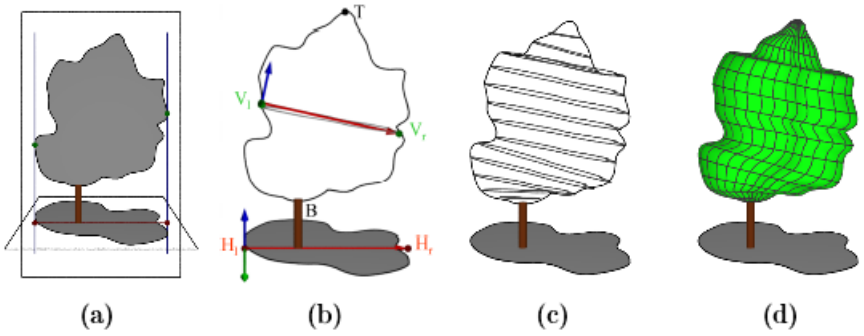


Figure 2.5: Extruded hull reconstruction. (a) Acquisition of a vertical and a horizontal profile. (b) Transformation of the horizontal profile into a section of the hull. (c) Computation of all sections (d) Mesh reconstruction.

2.1.1.4 Skinned envelope

The skinned envelope, proposed by Pradal et al. [Pradal *et al.* 2009], is inspired from skin surfaces [Woodward 1988, Piegl & Tiller 1997, Piegl & Tiller 2002]. The basic idea of this approach is a process of passing a B-spline surface through a set of vertical profiles given for any angle around the z-axis (see figure 2.6).

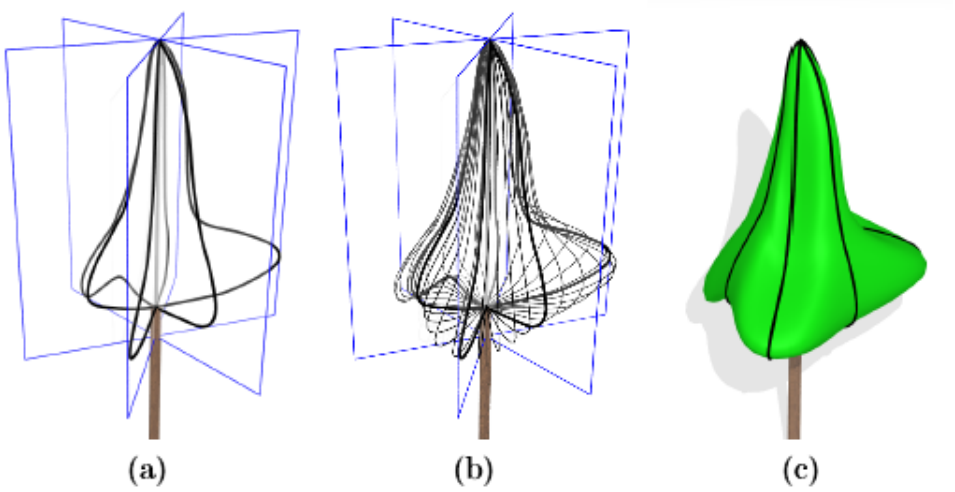


Figure 2.6: Skinned envelope reconstruction. (a) The user defines a set of planar profiles in different planes around the z axis. (b) Profiles are interpolated to compute different sections. (c) The surface is computed. Input profiles are iso-parametric curves of the resulting surface.

Given a set of profiles $\{P_k(u), k = 0, \ldots, K\}$ positioned at angle $\{\alpha_k, k = 0, \ldots, K\}$ around the z axis. All these profiles $P_k(u)$ are assumed to be non-rational B-spline curves with common degree p and number n of control points $P_{i,k}$. A variational profile Q can be thus defined that gives a section for each angle α around the rotation axis (see figure 2.6.b). It is defined as

$$Q(u, \alpha) = \sum_{i=0}^{n} N_{i,p}(u) Q_i(\alpha) \tag{2.4}$$

where the $N_{i,p}(u)$ are the pth-degree B-spline basis functions and the $Q_i(\alpha)$ are a variational form of control points. $Q_i(\alpha)$ are computed using a global interpolation method [Piegl & Tiller 1997] on the control points $P_{i,k}$ at α_k with $k \in [0, K]$. For this, let q be the chosen degree of the interpolation such as $q < K$.$Q_i(\alpha)$ are defined as

$$Q_i(\alpha) = \sum_{j=0}^{K} N_{j,q}(\alpha) R_{i,j} \tag{2.5}$$

where the control points $R_{i,j}$ are computed by solving interpolation constraints that results in a system of linear equations:

$$\forall i \in [0, n], \forall k \in [0, K], P_{i,k} = Q_i(\alpha_k) = \sum_{j=0}^{K} N_{j,q}(\alpha_k) R_{i,j} \tag{2.6}$$

Geometrically, the surface of the skinned envelope is obtained by rotating $Q(u, \alpha)$ about the z axis, α being the rotation angle. Thus, it can be define as

$$S(u, \alpha) = (\cos \alpha Q_x(u, \alpha), \sin \alpha Q_x(u, \alpha), Q_y(u, \alpha)) \tag{2.7}$$

2.1.2 Compartment-based representation

Compartment-based approaches that decompose plants into two or more compartments are intended to model exchanges of substances within the plant at a global scale [Godin 2000]. The core of such approaches is the description of how substances transfer within the plant. A compartment can be various elements such as leaves, roots, fruits or wood with connections between one another. The topological descriptions of the plant architecture for these models are very simplified due to the small number of compartments defined. In some cases, additional compartments can be included in order to refine the modelling of element exchanges within the plant.

The first compartment models have been introduced to model the diffusion of assimilates in plants [Thornley 1969, Thornley 1972]. These models describe transport-resistance availability of substances between leaf and root compartments. In figure 2.7.a, a stem compartment can be added, for instance, to model the growth process of the stem and to take into account the consumption of assimilates in the diffusion process [Deleuze & Houllier 1997].

In figure 2.7.b, illustrates the water transport model for tree. In this model, soil-plant hydraulic continuum are divided into compartments. Therefore, plants are represented as a series of compartments at the interface between above and below ground parts. Each compartment has a specific hydraulic conductivity and the flow of water through the plant results from the difference in water potential between the surface of the leaves and the soil/roots [Sperry *et al.* 1998].

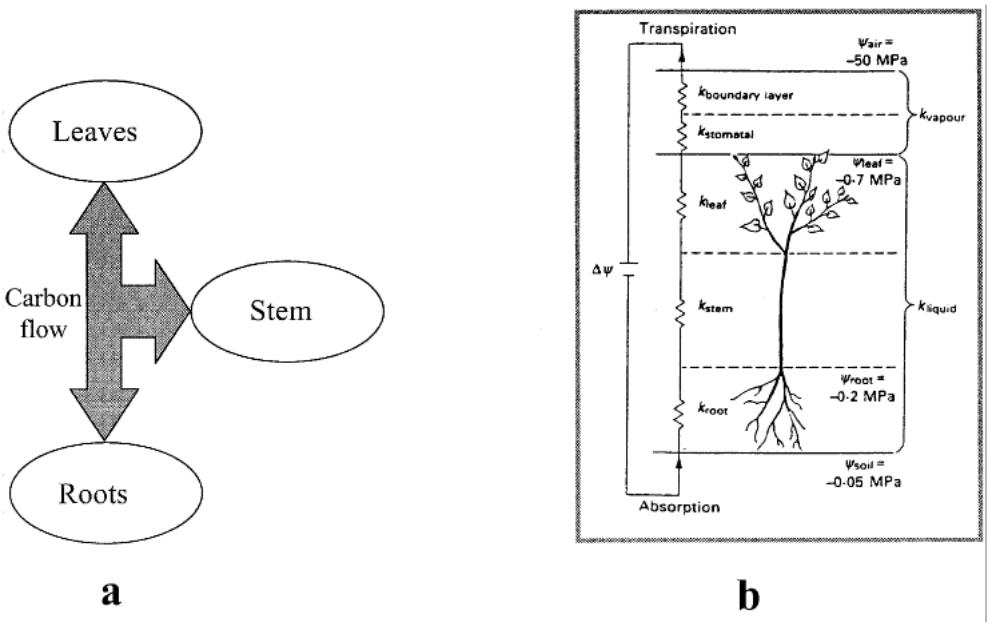


Figure 2.7: Compartment representations of plant architecture (a) in carbon partitioning models. Compartments are represented by different pools of carbon [Godin 2000]. (b) in water transport model, compartments are associated with the series conductance of hydraulically equivalent units (e.g. $k_{root}, k_{stem}, k_{leaf}$) [Tyree & Ewers 1991].

2.2 Detailed representations

To achieve realistic representations, we can classified the detailled plants representations that includes large amount of details as follow [Godin 2000]:

- *Spatial representations*, divides space in voxels and considers the ones occupied by the plant.
- *Topological representations*, describe the different components of the plant and their connections. This information expresses a notion of hierarchy among the components of a branching system.
- *Geometric representations*, describe the shape and spatial positions of the components of a plant.

2.2.1 Spatial representations

In spatial-based approach, plant components are not directly considered. The key idea is to represent the plant by the set of 3D voxels in which the plant is embedded. The space occupied by the plant can be subdivided into voxels with identical size σ as show in figure 2.8.a. An illustration of the spatial representation [Da Silva 2008] of an apple tree [Costes *et al.* 2003] can be seen in figure 2.8.b. Additionally, the voxels can also be attached with biological attributes characterising of plant components such as leaf density, optical properties and so on. A voxel representation V_σ can thus be defined as the list of occupied voxels I and optional attributes ρ:

$$V_\sigma = \{(i, \rho_i)\}_{i \in I}. \tag{2.8}$$

In the grid, as mention in Section 1.4, each voxel is identified by its indices (x_i, y_i, z_i). The spatial coordinates of each voxel can be defined as $(x_i.dx, y_i.dy, z_i.dz)$, where dx, dy and dz are constants.

Voxel-based representation of plants have been used in several works, e.g. [Greene 1989] describe the growth processes of plant in voxel space, [Sinoquet & Bonhomme 1992] introduce radiation interception within a voxel.

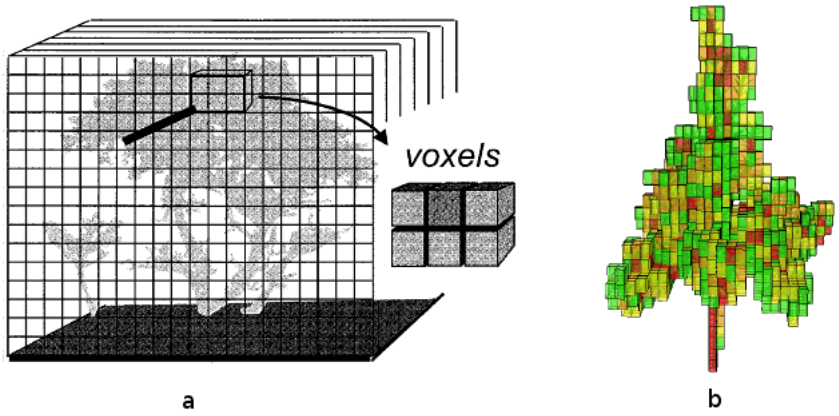


Figure 2.8: a. Representation of plant canopies using voxels with varying leaf densities. b. Spatial decomposition of an apple tree, cell colors depends on enclosed plant surface. From [Da Silva 2008].

In case of multiscale geometric representation, voxels can be further subdivided into smaller ones. The size of voxels varies according to the local irregularity of the object. Given V is a multiscale geometric representation with m scales. V can be thus formalized as a set of m nested voxel representations where the different voxel sizes are multiples of one another [Da Silva 2008]:

$$V = \{V_{\sigma_s}\}_{0 \leq s \leq m} \quad | \quad \forall s = \begin{cases} dx_s = k.dx_{s+1} \\ dy_s = k.dy_{s+1} \\ dz_s = k.dz_{s+1} \end{cases} , k \in N^{+*}. \tag{2.9}$$

An octree, as mention in Section 1.4, obtained by dividing voxel size by 2 at each scale is a simple version of multiscale spatial representation, see figure 2.9. As a hierarchical structure, the basic advantage of octree is that they do not need to be fully developped at all points to their full depth which thus reduce storage. The octree have been broadly used particularly in computer graphic applications. For example, [Sillion 1995] promoted the use of this structure for rendering and lighting simulation. It can also be generated from photographs of different view of the same object [Reche-Martinez *et al.* 2004, Phattaralerphong & Sinoquet 2005, Neubert *et al.* 2007] for visualization of reconstruction purposes.

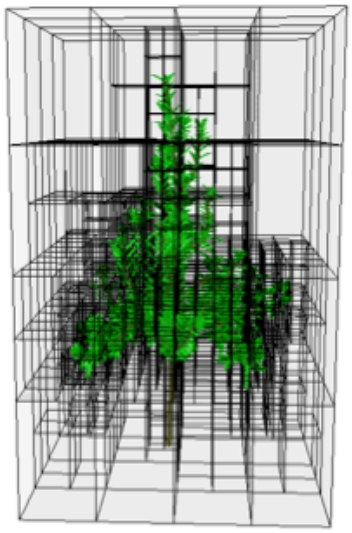


Figure 2.9: Octree representation of an apple tree. The object geometry is approximated by voxels whose sizes are locally adapted to its irregularity [Da Silva 2008].

2.2.2 Topological representations

Topological representations describe the organization of connections between various organs. These representations are the basis of many structure/function model of transfer of substances, plant growth, or plant architecture measurement. In computer graphics applications, [Honda 1971] initiated the use of topological representation of plant architecture. He demonstrated that complex crown shapes could be obtained using a limited number of geometric parameters and that plant architecture is very sensitive to change in these parameters [Godin 2000]. More elaborated models were introduced by ([de Reffye *et al.* 1988, Prusinkiewicz & Lindenmayer 1990, Prusinkiewicz *et al.* 1994]), who use the topological representation to obtain realistic rendering of 3D plants. All these applications have a common underlying structure, namely a *tree graph*. We will describe in this section the formalism used to represent plant as tree graph.

2.2.2.1 Representation at one scale

A plant can be decomposed as a set of components. The components of a plant and their adjacency relation can be represented by a binary relation defined on the set of components, i.e. a tree graph.

Let us define a graph as follows

Définition 1 *A graph G is a pair $G = (V, E)$ where*
V denotes a finite set of vertices,
E denotes a finite set of edges, each edge being represented by a pair of vertices, $E \subseteq V \times V$

Définition 2 *A tree graph is a simple directed graph without cycles.*

Définition 3 *A rooted tree is a directed tree which has a vertex r connected to all other vertexes by a unique path having r as its origin.*

The vertex r is called the root of the tree. Vertexes without any successor are called leaves of the rooted tree.

Godin and Caraglio [Godin & Caraglio 1998] introduced a global formalism to represent plant organization using trees. In figure 2.10.a,b shows a tree graph in which each branch of the plant is represented by a vertex and connections between branches are represented by edges between vertices. The edges are distinguished into two types. A '<' (precedes) indicates the connection between two components that have been created by the same apical meristem. When two components have been created by different apical meristem, the connection is noted by '+' (bears). Each vertex in the tree graph can also add additional information associated with plant organ. This information may correspond to the position in space of an organ, its geometry, or other characteristics of the organ. The resulting representation is called an *augmented tree graph*. An illustration is given in figure 2.10.c,d.

An axial tree [Prusinkiewicz & Lindenmayer 1990] is a special type of rooted tree, see 2.11.b. At each of its nodes, at most one outgoing *straight* segment is distinguished. All remaining edges are called *lateral* or *side* segments. The following definition is from [Prusinkiewicz & Lindenmayer 1990].

Définition 4 *A sequence of segments is called an axis if:*

- *the first segment in the sequence originates at the root of the plant or as a lateral segment at some node,*
- *each subsequent segment is a straight segment and,*
- *the last segment is not followed by any straight segment in the plant.*

An axis and all its descendants define a axial sub-tree. Axes have order, the one originating at the root has order 0. Any axis born by a n-order parent axis has order $n + 1$. The order of an axial sub-tree is equal to the lowest-order of its constituting axes. Note that it exists an equivalence between an augmented tree graph and an axial tree as illustrated in figure 2.11.

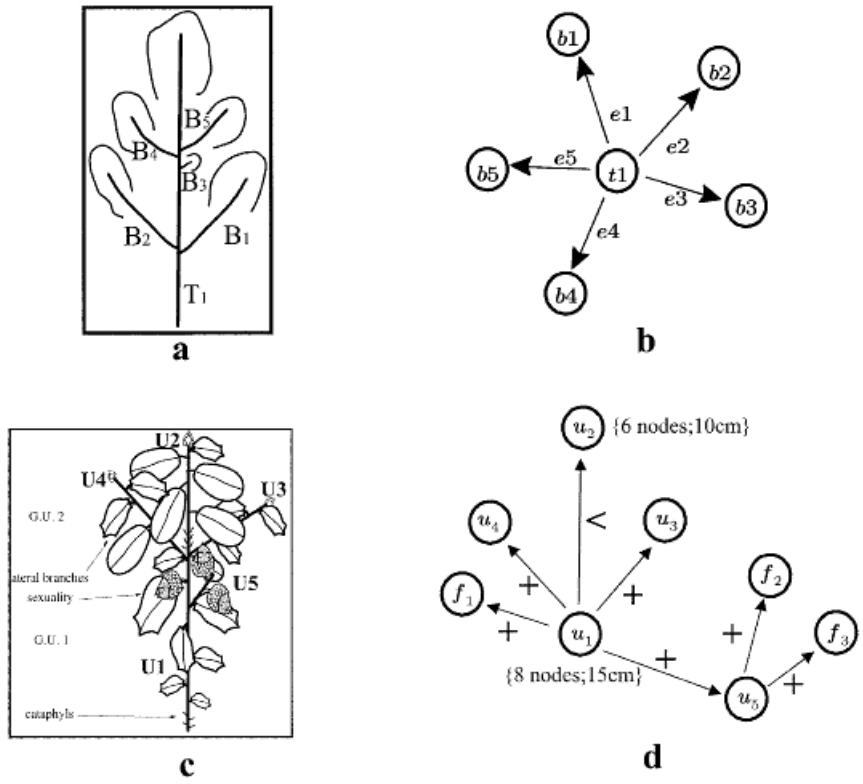


Figure 2.10: a. A tree: considered as a set of branches. b. the tree graph representation of its branch topology c. an oak tree branching system described in terms of growth-units and d. its corresponding augmented tree graph. [Godin 2000]

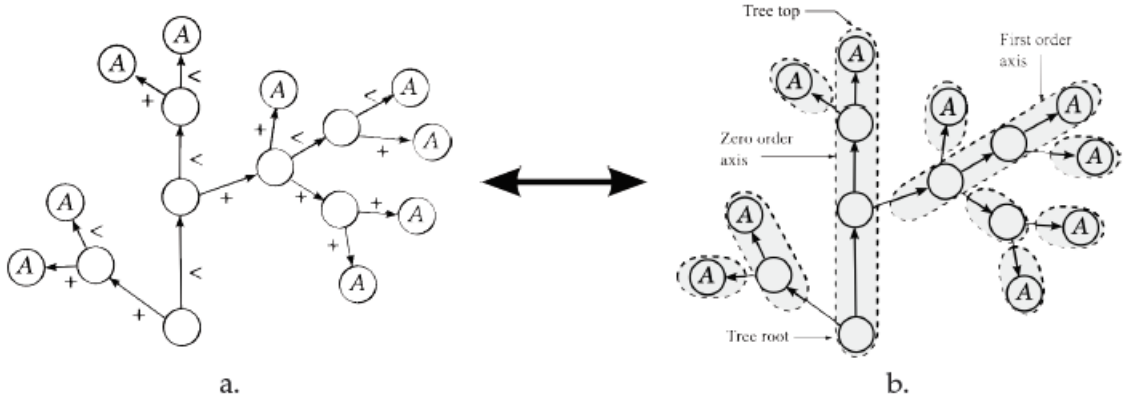


Figure 2.11: Equivalence between an augmented tree graph, a., and an axial tree, b. [Godin & Caraglio 1998]. Vertices in one representation are edges in the second and vice versa. Letter A denotes terminal components called apex. Tagging edges with connection information and grouping components into axis are equivalent.

2.2.2.2 Representation at multiple scales

In different systems for the simulation of plant growth, the modeling problem of morphological or physiological aspects at different scales of time and space leads to take into account the topological structure of the plant at different scales. Moreover, in the analysis of plant architecture, different levers of organization should be considered for a better understanding of the development of the plant [Sinoquet *et al.* 1997a]. Godin and Caraglio [Godin & Caraglio 1998] have formalized the notion of multi-scale structures of plants and have studied their mathematical properties. These representation model of plant architecture, based on tree-graphs, is called *MTG : Multiscale Tree Graph*. Each scale in a MTG is associated with the choice of a decomposition unit. In figure 2.12, the plant is decomposed into three different scales corresponding to three units of decomposition: axes, growth units and internodes. At each scale, the structure of the plant is represented by a tree. All these trees are part of a MTG making explicit the relationships between the components of decomposition. Based on multi-scale structures, we will define multiscale tree graphs.

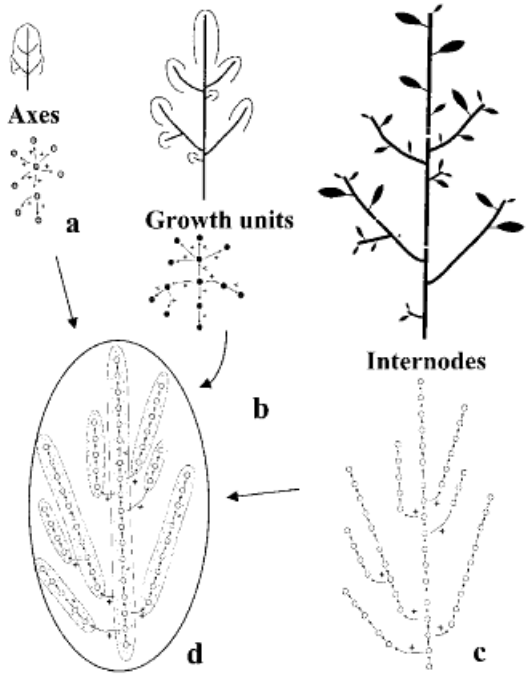


Figure 2.12: A tree at different scales of perception. (a) axis scale, (b)growth unit scale, (c) internode scale, (d) corresponding multiscale tree graph. [Godin & Caraglio 1998]

Quotiented tree graph

Définition 5 *A quotiented graph G is a triple (H, W, π) where:*

- *H = (V,E) is a directed graph, called the support of G,*
- *W is a set of vertices,*
- *π is an onto mapping from V to W, call the quotiented map.*

The graph H represents the topological structure associated with the finest modularity, W represents the set of constituents of the coarse modularity and the onto mapping π represents the decomposition relation: it defines how every constituent z of the coarse modularity (i.e. of W) is decomposed into a set of finer constituents, $\pi^{-1}(z)$, see illustration between *scale 3* and *scale 2* in figure 2.13. The vertex $\pi(x)$ is called the complex of x and reciprocally, x is a component of $\pi(x)$. The mapping π induces a partition Π_G over the set of vertices V:

$$\Pi_G = \{\pi^{-1}(z) | z \in W\}. \tag{2.10}$$

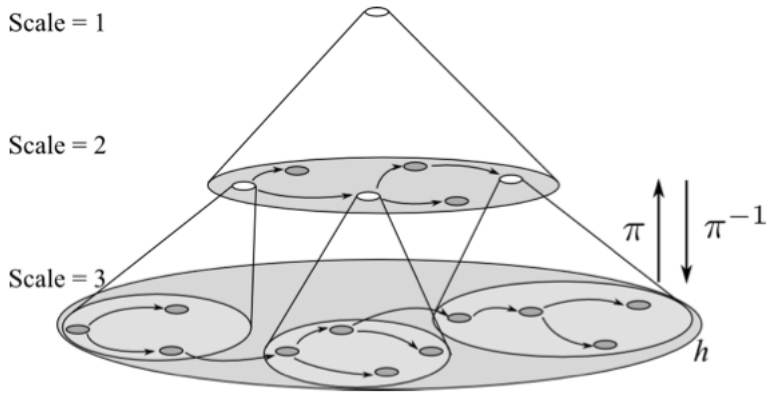


Figure 2.13: Illustration of a multiscale tree graph. The decomposition of all vertices is not shown for readability reasons [Ferraro 2000]

We shall denote $\Pi(x)$ the block of Π_G containing the vertex x of V. The partition Π_G is thus a representation of the coarse modularity in terms of the elements of the fine modularity. To ensure coherence between the different scales representations of a plant, the quotient operation must ensure that, if there exists an edge between two components at scale n and if the complexes of this two components are different, then there must be an edge between the two complexs at scale $n-1$ [Godin *et al.* 2005]. Connection between two modules results from the connection between two of their components.

Multiscale tree graph (MTG)

Quotiented graphs can be generalized by recursively applying new quotient operations on the successive quotient graphs obtained from an initial graph. Applying

several consecutive quotient operations on an initial object rapidly leads to collapse the entire object into one single vertex. The resulting structure is a pyramid whose bottom is made of vertices at the highest scale and whose top is made of the vertex at the most macroscopic scale representing the entire plant.

Définition 6 *The following definition formalizes these notions to define recursively multiscale graph [Godin & Caraglio 1998]:*

- *a simple graph G is a multiscale graph. The set of vertices of the simple graph G, denoted by V(G), defines the set of vertices of the multiscale graph.*

- *if H is a multiscale graph, then G = (H, W, π) is a multiscale graph as:*

- *V(H) is a set of vertices,*

- *π is an mapping from V(H)onto W, denoted π_G in case of confusion.*

A simple graph G is called *terminal*, otherwise it is called *recursive*. For a recursive multiscale graph $G = (H, V, \pi)$, H is called the support of G. In this case, the first three components of multiscale graph, (H, W, π), are similar to the components of a quotiented graph except for the fact that H itself is now, more generally, a multiscale graph instead of a simple graph. From the definition, we can see that a multiscale graph G can be developed recursively until a simple graph is obtained:

$$\begin{aligned} G = G_1 &= (G_2, V_1, \pi_1) \\ G_2 &= (G_3, V_2, \pi_2) \\ &\dots \\ G_{m-1} &= (G_m, V_{m-1}, \pi_{m-1}) \end{aligned}$$

The sequence $\{G_1, G_2, \dots, G_m$, where $G_1, G_2, \dots, G_{m-1}$ are recursive multiscale graphs and G_m is a terminal multiscale graph, is called the *development* of G, Because of their constructive definition, multiscale graphs have always finite developments. If H belong to G, we say that H *is finer* than G

2.2.3 Geometric representations

Geometric representation consists of decomposing plants into organs such as trunk, branches, leaves or different types of growth units, and considering their shapes and spatial organisation. In this section, modeling techniques used to build the geometry of plants are presented.

2.2.3.1 Representation at one scale

The shape of plant component is modeled by a geometric primitive. Two main types of elementary geometric models can be distinguished as follow:

- *parametric models*: define a curve, surface, volume using a finite number of parameters. Once a parameter was modified, the model will update to reflect the modification. Some examples of parametric models are Bezier surface, NURBS, cylinder, generalized cylinder, etc.

- *implicit models*: use a number of equations to describe a model. Examples of implicit models are cone, sphere, implicit surfaces, etc.

Figure 2.14, illustrates examples of basic geometric models generated from the PlantGL [Pradal *et al.* 2009].

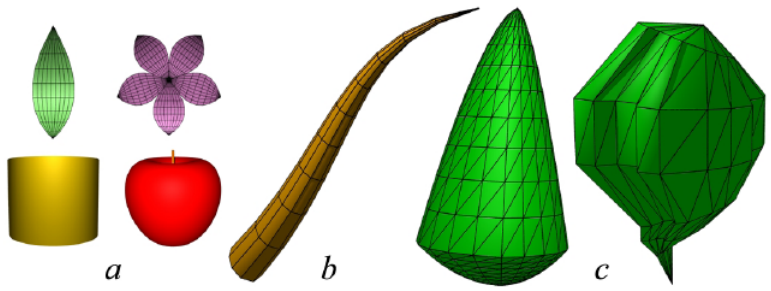


Figure 2.14: Example of basic geometric model (a) symbols to represent entities of a plant (cylinder, Bezier surface, etc.) (b)generalize cylinder represent a branch (c) two types of envelops: asymetric and extruded envelops. [Boudon 2004]

Branching systems

A. Simple approaches

The simplest way for modeling the geometry of branching structure is the representation of branch segments as 3D cylinders. De Reffye et al. [de Reffye *et al.* 1988] has defined a model of growth of the plant architecture using polygonal cylinders with various dimensions and orientations and in different positions. [Kawaguchi 1982] proposed a process for modeling corals, horns, etc., from circular polyhedron, see figure 2.15.

Unfortunately, these techniques do not properly capture the geometry of branch junctions, since some gaps or discontinuities between elements may appear [Boudon *et al.* 2006]. The subsequent approaches have been proposed to solve this problem.

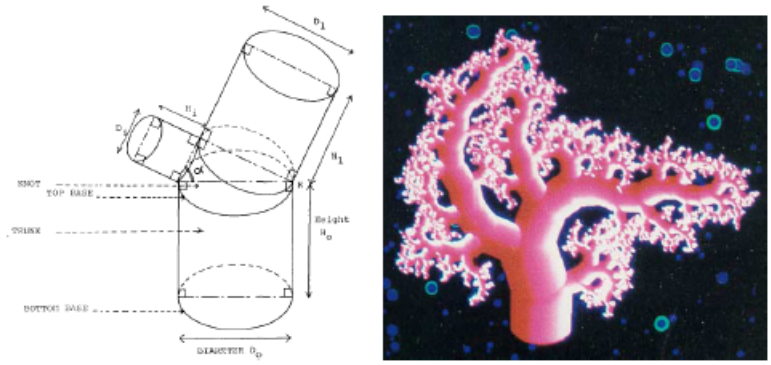


Figure 2.15: Left: Branching point geometry with circular polyhedron. Right: Example of branching structure. [Kawaguchi 1982]

B. Generalized cylinders

To alleviate the problem of branching points, [Bloomenthal 1985] proposed to use B-splines (see [Piegl & Tiller 1997]) for modeling the geometry of branch junctions. His approach start from a skeleton specified as a list of 3D points and a list of connections between those points. The branching pattern is generated recursively from the cubic spline interpolation of each set of n edges (n+1 nodes).

The surface of each branch is then considered a generalized cylinder. The generalized cylinder provides a useful tool for the modeling of tubular shapes. Given a 2D cross section curve of varying radii and a 3D skeleton curve, the generalized cylinder is defined as the sweep surface of the cross section curve moving along the skeleton curve. To polygonize the surface, a finite number of cross sections are evaluated along the skeleton curve and connected together.

To handle the problem of branching points, various generalized cylinders are connected using several parametric surface forming a 'saddle'. A saddle surface is a smooth surface containing one or more saddle points. As illustrate in figure 2.16-left, a spline (shown in yellow,4) is constructed between disk 2a and disk 2b passing through the saddle point S.

Bark is simulated with a bump map built from digitized bark and leaves photographs are textured onto simple quad surfaces. Figure 2.17 illustrates the visual quality of the result.

C. Implicit surfaces

The problem of representation of the branches and their connections can also be naturally addressed using implicit surfaces [Bloomenthal 1995]. An implicit surface S is defined as the set of points $P = (x, y, z)$ at which the value of a field function $F(x, y, z)$ is equal to 0. These surfaces can be built from skeletons. Such skeletons can be any geometric primitive for which one can calculate the distance to a point. Each skeletal primitive s_i contributes to F with a field function f_i which is decreasing with the distance to this element. The general equation of an implicit surface is thus

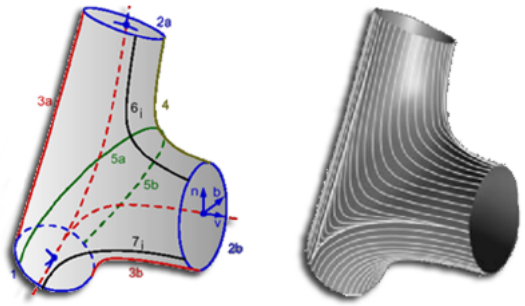


Figure 2.16: Representation of the branching points with a ‘saddle‘ surface, built by crafting several parametric surfaces. On the right, texture parametrization [Bloomenthal 1985].

Figure 2.17: The mighty maple. [Bloomenthal 1985]

defined as:

$$S = \{P = (x, y, z) \in \mathbb{R}^3, F(P) = \sum_i f_i(P) = 0\} \tag{2.11}$$

However, the blend at the branching points can be affected by the way of each f_i varies.

Bloomenthal and Wyvill [Bloomenthal & Wyvill 1990] proposed to used the *distance surfaces* which f_i are defined as decreasing C^1 continuous functions G of the distance to s_i (defined as the distance to the closest point on s_i):

$$f_i(P) = G(d(P, s_i)) \tag{2.12}$$

An example of implicit surface defined by the skeleton is given in figure 2.18.

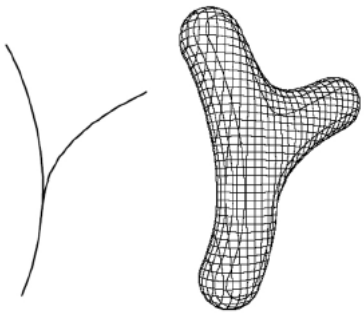


Figure 2.18: A skeleton and the corresponding implicit surface. [Bloomenthal & Wyvill 1990]

If, however, two segments intersect skeletons on the cross, implicit surface generated by the previous method create a greater thickness at the junction of two segments, called bulge (see figure 2.19).

To avoid bulging, Bloomenthal and Shoemake [Bloomenthal & Shoemake 1991] introduced *convolution surfaces*, and defined their field functions as:

$$f_i(P) = \int_{s_i} e^{-\|P-X\|^2/2} dX \tag{2.13}$$

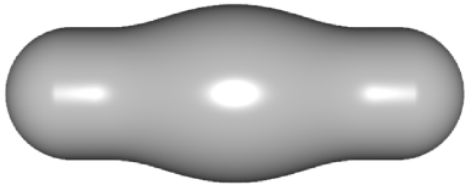


Figure 2.19: Bulge when two line segments with coincident end points are blended. [Galbraith *et al.* 2004]

Scene graph for plants

To organize the geometrical representation of a plant branching structure, a computer tree structure can be used. For this, a *p-scene-graph* has been formalized in PlantGL [Pradal *et al.* 2009]. A p-scene-graph is organized around a tree graph as described in section 2.2.2. A first set of nodes, named structural nodes, represented the different components of a plant. The topological relationships between these nodes express physical connections between plant components. In figure 2.20, a structural node (orange color) is connected to other structural nodes with two types of edges: '+' for branching and '<' for succession between nodes. Additionally, each structural node can also be connected with transformation, geometry and appearance nodes that define its geometric representation.

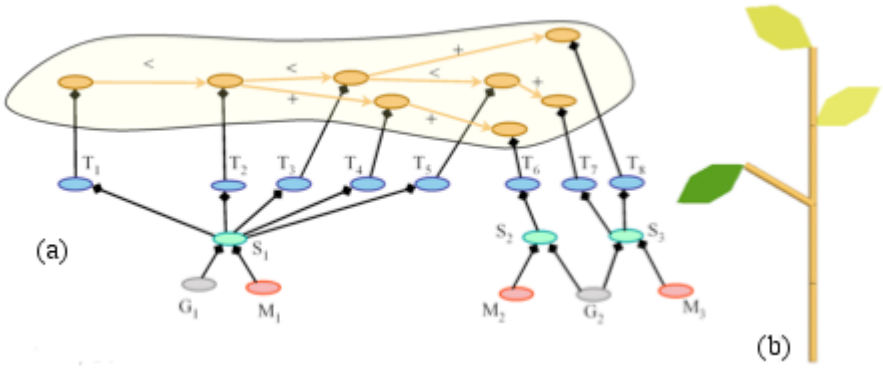


Figure 2.20: A p-scene-graph. (a) The scene-graph with structural nodes in orange, transformation in blue, shape in green, appearance in red and geometry in gray. (b) The corresponding geometric model. [Pradal *et al.* 2009]

2.2.3.2 Representation at multiple scales

Multiscale scene graph for plant

The p-scene-graph structure can be extended by introducing a multiscale organization in the structural nodes, which makes it possible to augment the multiscale tree graph (MTG) used in plant modelling [Godin & Caraglio 1998] with graphical informations.

The multiscale p-scene-graph for plant is built by combining the different views of the same plant object on the groups of their representation. For example, a simple ramified system can be represented at two different scales in a p-scene-graph: the first scale represents the internodes scale, and the second scale represents branch scale, see figure 2.22.

This structure is used for instance to design hierarchically a tree [Boudon *et al.* 2003]. Figure 2.23 shows the hierarchical structure of a plant induced by its branching order. Each node of the graph is place-holders for the geometrical primitive of the plant at different levels of the hierarchy, and thus at different levels

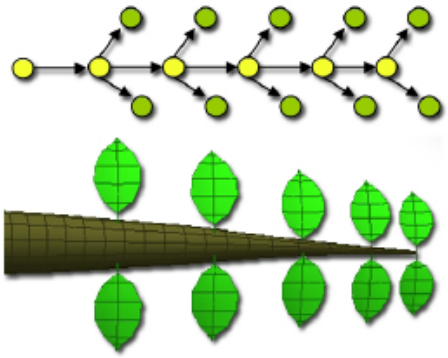


Figure 2.21: A branch consists of five inter-nodes (represented by yellow nodes in the graph) and two leaves (represented by green nodes in the graph). The relationship between the orientations of the geometric models of the different entities can be translated as constraints intra-scale carried by the topological graph. For the leaves, these constraints are expressed in terms of their angles of insertion and phyllotaxis of the branch. [Boudon 2004]

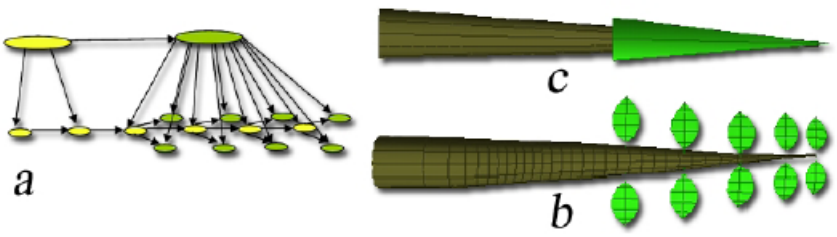


Figure 2.22: (a) A multi-scale representation of a simple ramified system. (b) the representation at inter-nodes scale. (c) the representation at branch scale. [Boudon 2004]

of detail.

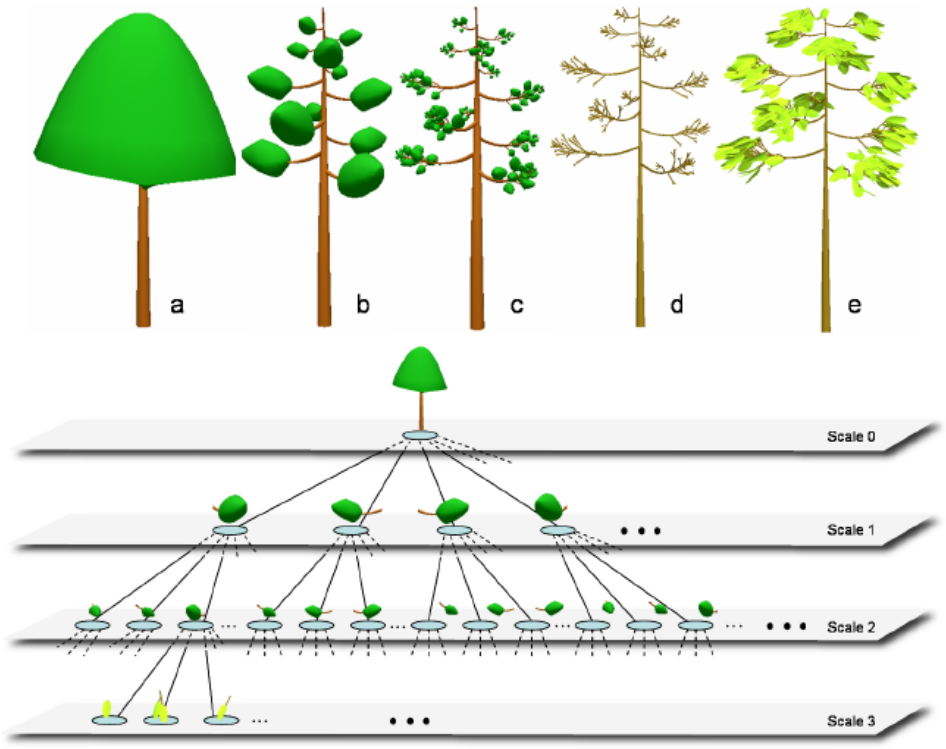


Figure 2.23: Top: approximate representations of a tree structure at scales 0 to 3 (a-d), and the final tree model (e). Bottom: the corresponding multiscale p-scene-graph. [Boudon *et al.* 2003]

Point based representation

Since plant scenes are usually complex in details, it is difficult to efficiently manipulate and render their geometry. Point based representations are an approach for efficiently rendering complex geometric objects. Unlike various modeling techniques, i.e., implicit surfaces, NURBS or subdivision surfaces, this representation is based on point primitives without explicit connectivity. It seems to be appropriate for plant scenes because collections of discrete points can convey the shape of distant plants naturally. This approach is based on the observation that in increasingly complex scenes triangles become smaller than a single pixel [Boudon *et al.* 2006]. Since points are merged easily, the degree of detail of any complex geometry can thus be fluently adapted by adding or removing single points. A hierarchical data structure allows to smoothly reduce the geometrical representation to any desired number of primitives.

The forest model of Reeves et al. [Reeves & Blau 1985] was probably a source of inspiration for many works on point based representation. Instead of concentrating on the structural detail of individual plants, this method focused on visual results of

forest environment and not on the specific details of actual botanical data. The data structure for the model is a tree in which each node describes a branch segment. To generate complex scene from set of simple primitives, the main trunk of each plant is first constructed and then recursively generate sub-branches. When all plants are completely constructed, the algorithm renders the model, see figure 4.28.

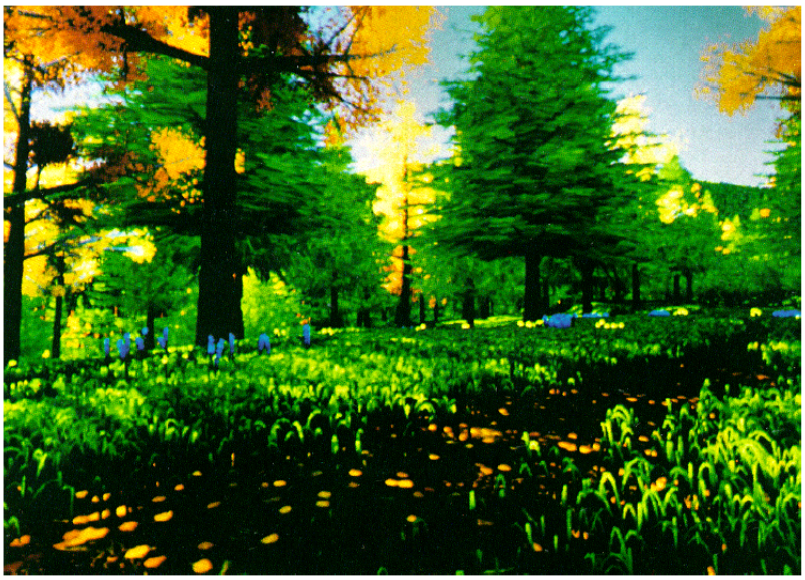


Figure 2.24: Generate the forest scene from point based representation. [Reeves & Blau 1985]

Later, Weber et al. [Weber & Penn 1995] proposed a representation in the level of detail of plants that combine polygons and points/lines representations: tree skeleton as lines and leaf polygons as points. The geometry (polygons, points and lines) is stored in vertex arrays on the GPU. In this approach, the geometry of an element (branches and leaves) disappears depending on the distance from the object to the camera and the resolution of the image, see figure 2.25 (top). Therefore, it could lead to important visual artefacts. Deussen et al. [Deussen *et al.* 2002] addressed this issue by merging disappearing elements in larger primitives (points or lines) to better convey the shape in distant views. Long and thin parts of plants such as leaves or branches are represented by lines, while more compact objects are approximated by points, see figure 2.25 (bottom).

Conclusion

In this chapter, a wide variety of models for representing plant architectures have been outlined, ranging from global to detailed representations. The global representation which reduce to a minimum the plant representation are often concise. This allow the modeller to design simple models, i.e. models with a small number of parameters, which in turn favours a biological interpretation of the model structure.

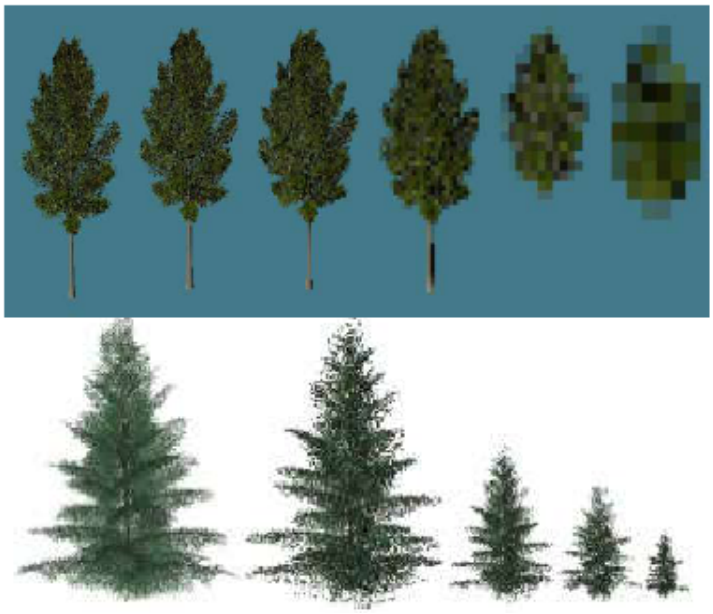


Figure 2.25: Representation tree skeleton as lines and leaf polygons as points: with distance (top) [Weber & Penn 1995] and with progressively re-interpret the tree (bottom) [Deussen *et al.* 2002].

By contrast, detailed representations of plant architecture decompose the architecture into basic components. They potentially allow the user to address different scales of representation.

Chapter 3

A review of acquisition and reconstruction methods of plants model

Natural plants remain one of most difficult kinds of object to acquire and model due to their complex geometry and wide variation in appearance [Quan *et al.* 2006]. The relative merits and difficulties associated with several measurement techniques are described in Chapter 1. Some details on generals reconstruction techniques from laser scans are also presented. However, because of the complexity of the plant shapes, these techniques are difficult to apply directly. In this chapter, the existing methods to reconstruct 3D models of plants are reviewed. In section 3.1, we first introduce approaches based on manual measurement techniques. We then described methods based on freehand sketching in section 3.2, image-based approaches based on photographs in section 3.3. Finally, reconstruction techniques for reconstructing tree structure from laser scans are described in section 3.4.

3.1 3-D Digitizing

Over the last decades, 3-D digitizing has been used for measuring plant architecture. With such approach, the 3D coordinates of each organs of a plant is individually captured by a user, requiring extensive user interaction. According to Moulia and Sinoquet [Moulia & Sinoquet 1993], two main approaches exist: contact digitizing and non contact digitizing. Both approaches will be described in the next section.

3.1.1 Contact digitizing

The first contact 3-D plant digitizer, built by Lang [Lang 1973], used articulated arms to contact the plant point and recorded the coordinates of the pointer. This elaborate device based on high precision potentiometers directly determines 3-D distributions of plant parts. The coordinates are computed from length and angle where rotation angles are recorded from potentiometer resistance values. However, such systems require electricity and are not convenient to carry around in the field. Later, Takenaka et al. [Takenaka *et al.* 1998] introduced a device, called *pocometer* (POlar COordinate METER).The pocometer consists of two protractors and an arm with a tape measure. Distance to a point of interest is measured with the tape measure and the protractors are used to measure the azimuth and zenith angles, see

figure 3.1. This device is easier to carry around in the field and does not require eletricity. Accuracy of this device depends on distance between base and sample plants. The main problem is that some points may not be detected due to insufficient degree of freedom of the device or intricate vegetation.

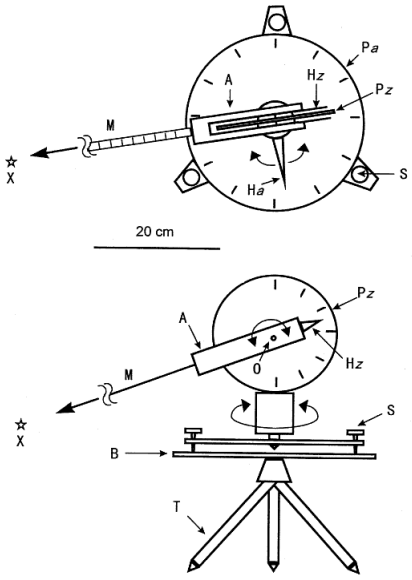


Figure 3.1: Pocometer: a device for measuring components of polar coordinates. *top*: view from above; *bottom*: side view. A, arm; B, base; Ha, hand for the azimuth; Hz, hand for the zenith angle; M, tape measure; Pa protractor for the azimuth; Pz protractor for the zenith angle; S, screws for level adjustment; T, tripod; X, the point to be measured. Arcs with arrowheads on both ends indicate rotation. [Takenaka *et al.* 1998]

The sonic digitizer [Sinoquet *et al.* 1991, Watanabe *et al.* 2005] are based on the principle of triangulation method. It consists of a probe with sonic emitters and triangular detector array with microphones. The coordinates of the measured point are computed from the speed of sound between emitters and detectors. Therefore, calibration is necessary to correct for differences in temperature and humidity of the air. The main problem with sonic digitizers are that they are very sensitive to wind fluctuations in the field. Moreover, spatial coordinates in the inner part of dense canopies cannot be recorded because vegetation elements between the pointer and sound receptors may disturb sound propagation. Hence, they should be used only in the laboratory [Godin *et al.* 2005].

Magnetic digitizing device, such as FASTRAK magnetic 3D digitizer (Polhemus, Colchester, VT, USA, see figure 3.2), were succesfully used with plants [Sinoquet & Rivet 1997, Rakocevic *et al.* 2000, Evers *et al.* 2005]. This device includes a pointer and magnetic signal receiver, allowing to record the 3D spatial coordinates and the orientation angles of the pointer located in a magnetic field. To digitize the plant, the device generates a magnetic field around the

plant and coordinates are then computed by a central unit from the measurement of induced currents in coils included in the pointer [Polhemus 1993]. This system is suitable for plant digitizing. First, the magnetic fields are insensitive to the presence of the plant. Second, it allows the measurement of orientation angle of components with accuracy. Third, the active volume can be large, up to 80 m^3. This is probably the reason why it has been widely used in the last decade on a large range of canopy structures: trees ([Sinoquet & Rivet 1997, Sinoquet *et al.* 1997b, Godin *et al.* 1999, Rakocevic *et al.* 2000, Costes *et al.* 2003, Evers *et al.* 2005]), annuals ([Phattaralerphong & Sinoquet 2005]), and forage crops ([Sonohat *et al.* 2002]). However, in greenhouse environments the surrounding iron frames can disturb measurements.

Figure 3.2: FASTRAK system

The major advantage of contact digitizing is that the measured points can be recored directly with additional information about the corresponding organs. Thus, the identification of the plant components and topology can be annotated simultaneously. Softwares developed to help 3D plant structure acquisition were developped. Floradig ([Hanan & Wang 2004]) is able to drive both sonic and magnetic digitizers. 3A ([Adam *et al.* 1999]) makes it possible to record both plant topology according to MTG coding style and 3D coordinates from a magnetic digitizer, see figure 3.3.

3.1.2 Non contact digitizing

Contact methods are considered to be the most accurate way to describe plant architecture. However, they are also tedious, as they need an operator moving the pointer onto the vegetative surfaces. Therefore, they are generally suitable only for small plants with few leaves [Farque *et al.* 2001], because they are particularly time consuming when it is applied to larger plants.

Non contact digitizing can be used to obtain expert description of tree architecture. It also eliminates any possibility of damage to a plant and can be applied

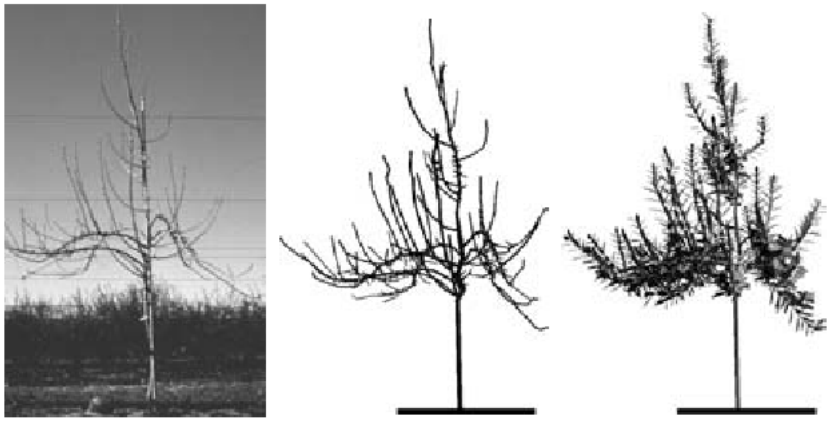


Figure 3.3: 3D digitizing of an apple tree (left) using the 3A software. 3D reconstruction of the branching system (middle) and 3D reconstruction of the leaves using allometric relationships between shoot length, number of leaves per shoot and leaf area and the AMAPmod software.

to digize leaves that are composed of soft material. These systems compute spatial coordinates using mainly the principle of triangulation (see section 1.1) or the parallaxe method. Parallaxe techniques use optical telemeters which are, for the most part, based on laser beam techniques. The main disadvantages of these techniques comes from occlusion between organs, for instance in dense canopies.

Note that partial manual digitizing can be combined with procedural generation and allometric rules to complete the structure, such as proposed by Sonohat et al. [Sonohat *et al.* 2006]. This enables significant time savings and could be used in virtual experiments requiring large numbers of replicates.

3.2 Modeling plant structure from sketching

Sketch based methods make it possible to interactively design plants using freehand sketches. The 2D strokes from user drawing are converted into 3D plant models by inferring hidden parameters. To design 3D plant geometries, user specifies the 2D appearance of the model directly used standard 2D input devices, e.g. a mouse or pen tablet, and the final 3D geometry is generated.

This approach was initiated by Okabe et al. [Okabe *et al.* 2006] that presented a system for quickly and easily designing three-dimensional models of botanical trees using freehand sketches and additional example-based editing operations. In their system, the user begins to model a 3D tree by sketching simple 2D strokes that represent branches or leaves. To distribute the branches in 3D around the trunk, the branches are projected horizontally to construct a 2D distance field (Figure 3.5, bottom). The orientation between each branch is then adjusted to be as large as possible using the 2D distance field (Figure 3.5). The lengths of branches are also adjusted so that their projection stay constant from the user point of view. However

to prevent too large branches, a 3D hull constructed from a 2D convex hull around the given sketch (figure 3.4) and limits the extension of the branches.

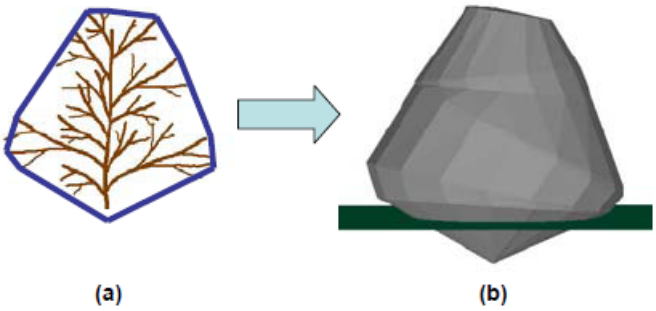


Figure 3.4: (a) A 2D convex hull of the original sketch. (b) The resulting 3D convex hull. [Okabe *et al.* 2006]

The authors assume that the user draws only branches that extend sideways. Therefore, the system automatically add branches that extend toward and away from the screen. These branches are generated as copies of previous branches defined by the user.

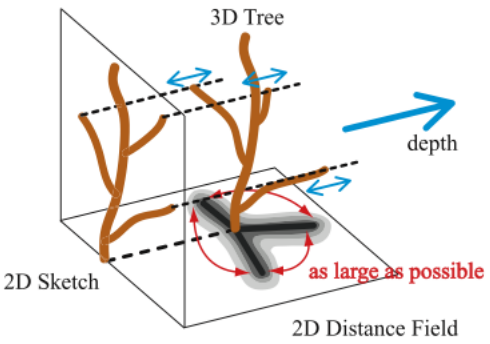


Figure 3.5: Computing the depth information for branches with the 2D distance field. [Okabe *et al.* 2006]

Wither et al. [Wither *et al.* 2009] extend this method with a multi-scale approach. The general idea is to sketch silhouettes of foliage at multiple scales instead of sketching each branch individually. At each scale, the user is only required to sketch a crown silhouettes. To infer 2D branching structure from a sihouette, tip of branches are first identified from geometric skeleton of the silhouette calculated using Chordal Axis Transform technique [Prasad 1997]. This tips are reconnected to the trunk drawn by the user according to a *branching angle function* that model the gradient of lateral branching angle of child branches along a parent trunk. The generated branches can then be corrected by the user. A sub-silhouette is then generated for each branches. The user can also resketch one silhouette to generate a new branching system at a finer scale. This is done until reaching the level of the

leaves. The branching system are distributed in 3D around the main branches by maximising a cost function that takes into account distance between branches and possibly botanical constraints such as phyllotaxy. When one exemplar of structure is made at each level, the style of structure and branches are copied to siblings at the same level.

The figure 3.6 illustrates this construction process. The user start to draw the crown silhouette of the whole tree at the scale of trunk (level zero (L0)). Child branches (L1) are automatically inferred from the silhouette shape and guidelines for sub-silhouettes (L1) are generated. After possibly making alterations through oversketching, the user selects an area to zoom into and progressively draws finer silhouettes until the level of leaves. Then, the last branch is placed in 3D using botanical-based distribution laws and the style is copied to the siblings at the same level.

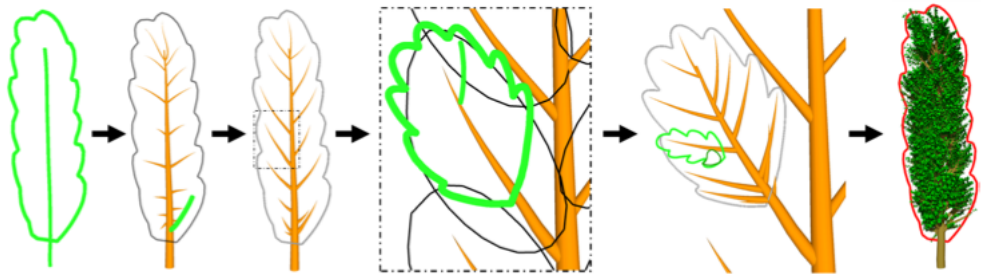


Figure 3.6: Creation of a 3D tree, by sketching successive silhouettes of foliage at different zoom factors, from the full tree to a leaf. Plausible branches and construction lines for the smaller-scale silhouettes are inferred from each sketch. When the user zooms out, each branching system transmits its style to the other systems at the same level and 3D is inferred, resulting in a full 3D tree. The latter can still be edited by over-sketching from arbitrary viewpoints. [Wither *et al.* 2009]

An approach inspired by a biological modelling algorithm considers the competition of branches for space as the dominant factor determining the form of trees. They allowed the user to control the form of trees by constraining the space in which branches grew. The process estimates the availability or quality of the space surrounding each bud and the optimal direction of shoot growth. To calculate these values, Runions et al. [Runions *et al.* 2007] and Palubicki et al. [Palubicki *et al.* 2009] present the procedural tree generation method, named *Space Colonization Algorithm (SCA)*. The key idea is an iterative addition of new elements (nodes) to the tree structure formed in previous steps. This process is guided by the proximity of points marking the availability of free space.

The steps of the procedure are shown in figure 3.7. It starts by defining the volume the crown of the tree. This volume can be defined by an envelop filled randomly with points or by a user that place points in space with a brush. Given the points, called *attraction points*, the tree skeleton is formed iteratively starting from a single node at the base of the tree. At each iteration, new nodes extend the

skeleton toward nearby attraction points. This process terminates when no nodes within the radius of influence of the remaining attraction points, or all attraction points have been removed. The skeleton nodes may be decimated and moved to create more smoothly curved limbs. Finally, the tree geometry is modeled using generalized cylinders centered on the axes of the skeleton.

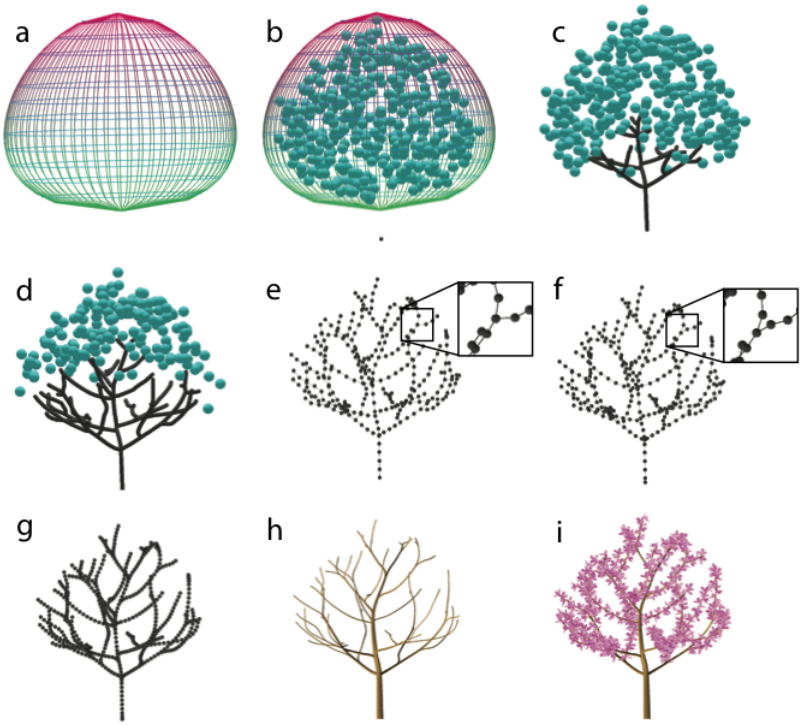


Figure 3.7: Key steps of the proposed method for generating trees. a) Specification of the input envelope; b) Placement of the initial tree node and attraction points in the envelope; c,d) Generation of the tree skeleton; e) Node decimation; f) Node relocation; insets show the modified branching angle; g) Subdivision; h) Construction of generalized cylinders; i) Addition of organs. [Runions *et al.* 2007]

3.3 Image-based plant modeling

The techniques presented in the previous section rely on the user skills to measure or depict a tree with sketches. More automatic methods were explored. In the past decade, image-based methods were used to acquire and reconstruct real 3D plants. Simple cameras were used for capturing the input images. A number of images of a tree are taken from different overlapping views around the tree. For instance, Shlyakhter et al.[Shlyakhter *et al.* 2001] took 4 to 15 images covering at least 135 degrees around the tree, Tan et al. [Tan *et al.* 2007] took 10 to 20 images with coverage between 120 and 200 degrees, and Quan et al. [Quan *et al.* 2006] used

between 30 to 45 input images of each tree. During the acquisition, the user has to be cautious to avoid as mush as possible appearance changes due to change in lighting, shadowing and wind. In this section, we described a number of steps that are required to reconstruct 3D plants from the images.

3.3.1 Image segmentation

Usually the input images contain background objects. Therefore, a segmentation of the pixel representing the tree has to be made. Shlyakhter et al. [Shlyakhter *et al.* 2001] segmented the images manually by outlining the foliaged tree region in a graphics editor. Reche-Martinez et al. [Reche-Martinez *et al.* 2004] and Neubert et al. [Neubert *et al.* 2007] extract the background and foreground using the alpha estimation algorithm proposed by Ruzon and Tomasi [Ruzon & Tomasi 2000]. This algorithm lets the user specifies pixels of the object and pixels of the background. The separation algorithm fills the uncertain regions with either opaque (foreground), transparent (background) or partly transparent pixels based on an interpolation which takes into account the image gradients. Therefore, the tree images can be clustered by color distribution, see figure 3.8. Quan et al. [Quan *et al.* 2006] and Tan et al. [Tan *et al.* 2007] used automatic color-based segmentation to extract 3D point cloud belonging to a tree from a given sequence of images. This approach, called quasi-dense matching, is to match the corresponding pixels in the two images by using the ZNCC (the Zero-mean Normalized Cross-Correlation) score together with a cross-consistency check [Lhuillier & Quan 2005].

Figure 3.8: Left: the original image. Middle: the regions specified roughly by the user as foreground and background. Right: the computed alpha-mask(note that they used black for opaque for clarity of the figure). [Reche-Martinez *et al.* 2004]

3.3.2 Reconstruction

An approach is to first reconstruct a crown envelop [Shlyakhter *et al.* 2001] as the visual hull. For this, the silhouette of the tree of each image are projected in space in the direction of the view corresponding to the image. This forms a set of cones and their 3D intersection gives the hull. A tree skeleton is computed as the medial axis

(described in section 1.5.2.1) of the hull. To complete the structure, the skeleton is exported as the axiom of an L-system that complete generate last orders of branches.

[Neubert *et al.* 2007] propose to estimate tree crown from loosely arranged images, to populate the crown volume with particles and to simulate their flows to generate plausible branches. To do this, tree density values for the voxels are first estimated from the opacity of corresponding pixels of the images. The particles are then placed randomly in the voxels in proportion to their density. The tree skeleton is reconstructed by first introducing a user defined branching structure, called attractor graph.

A particle p_i is represented by a position x_i, velocity $x_i^{'}$, and mass m_i. It moves under the influence of time-dependent forces represented by $f_i(x_i, x_i^{'}, t)$. The Newtonian law gives $f_i = m_i.x_i^{''}$ and $x_i^{''} = f_i/m_i$. Therefore, the velocity and position can be written in the form of coupled first order differential equations:

$$[x_i, x_i^{'}] = [x_i^{'}, f_i/m_i] \tag{3.1}$$

Particles close to each other are forced to join and subsequently move forward together. Each particles is also attracted by the attractor graph. For this, the closest point g_i on the graph is estimated for each particles. let $v_i = g_i - x_i$ be the vector pointing towards g_i. Additionally the tangential vector t_i is computed at position g_i. Using the normalized version of these two vectors $\bar{v_i}$ and $\bar{tv_i}$, the force that apply on particle i is equal to

$$f_i = h(d).\bar{v_i} + (1 - h(d)).\bar{t_i} \tag{3.2}$$

with $h(d)$ a blending function that depend on the distance of the particle to the graph $d = |v_i|$.

The particle tracing mechanism can be summarized as follow:

Procedure 1 ParticleTrace

```
Initialize particle position according to voxel density
while particles not at tree basis do
  for all particles p_i do
    determine forces f_i
    determine velocities and positions (Eq.3.1)
  end for
  determine nearest neighbor and join if close enough
end while
```

Tan et al. [Tan *et al.* 2007] proposed an automatic approach to generate branch structure from images. Once 3D point cloud belonging to the branches and leaves have been extracted from images using color and position, visibles branches are reconstructed. For this, a method similar to the one proposed by Xu et al. [Xu *et al.* 2007] described in section 3.4 is used. To complete the structure in particular in occluded area, new branches can be inferred using predefined branching

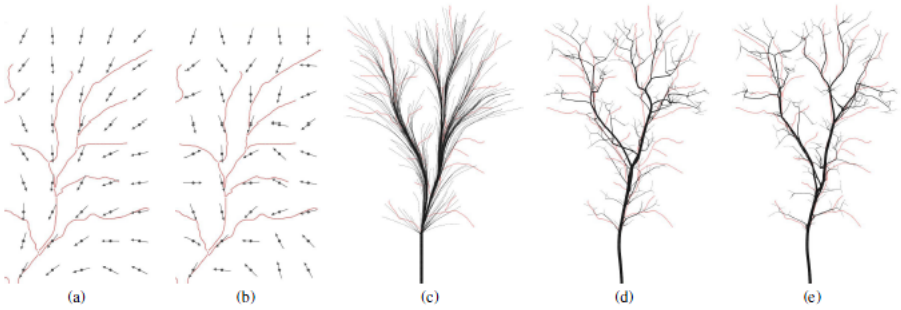


Figure 3.9: (a) Direction field for a given graph and linear blending function $h(d)$, the vectors close to graph segments point down the graph; (b) for a constant blending function, the vectors point at a given angle towards the nearest segment; (c) result for linear blending without particle attraction; (d) result for constant blending including particle attraction; (e) result for linear blending including particle attraction. [Neubert *et al.* 2007]

patterns determined from visible branches. These patterns are duplicated to populate the volume and reach the leaves. The position of each leaf segment is determined either by its closest 3D point or by its closest branch segment, see figure 3.10.

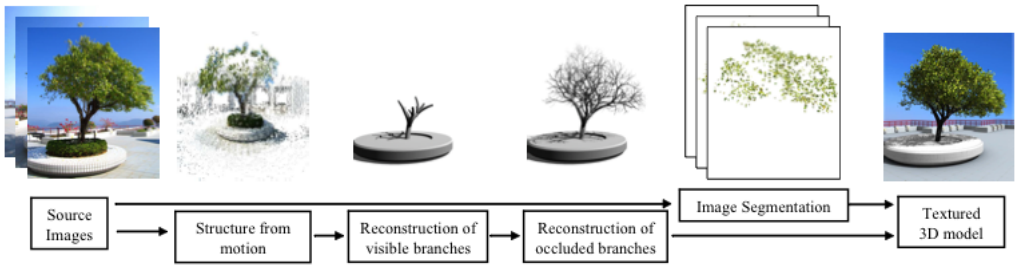


Figure 3.10: Image-based tree modeling system proposed by [Tan *et al.* 2007]. It consists of three main parts: image capture and 3D point recovery, branch recovery, and leaf population.

Tan et al. [Tan *et al.* 2008] introduce a simple sketching method to generate a realistic 3D tree model from a single image. This approach rely on user-drawn strokes to guide the synthesis, where the input is a single photograph. The user sketch the trunk and basic branches or the outline of the crown. Trees are then generated by image matching and randomization of existing models from a database.

3.4 Reconstructing plant model based on 3D-points

More recently, 3D laser scanner, described in section 1.2, make it possible to capture rapidly a depth image of plants and possibly their appearances (i.e. color). The additional depth information associated with each pixel of the scanned image makes

it possible to recover the position of the plant structure in 3D and some of its optical properties [Yan *et al.* 2009].

The first approach to reconstruct tree structures using scanning technology is proposed by Gorte and Pfeifer [Gorte & Pfeifer 2004]. In this approach, the sequential thinning method of [Palágyi *et al.* 2001] is used to skeletonize scanning points obtained from scanner. The points data is first transformed into a 3D-raster (corresponding to a grid of voxels). Skeletonization is done by iteratively removing voxels from the outside of objects. At every iteration, each boundary voxel is tested against a set of topology preserving conditions and possibly removed. The remaining voxels are then connected as a skeleton graph. Dijkstra's algorithms finds the shortest route to the root voxel from all other nodes (voxels) in the graph. The branches can be then identified by marking junction nodes that appear as a parent more than once, as illustrate in figure 3.11.

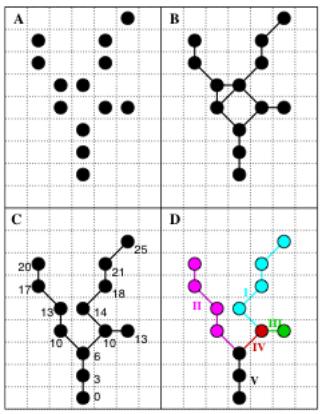


Figure 3.11: 2D illustration of skeleton segmentation. A) input skeleton; B) graph nodes and edges; C) spanning tree with shortest distances; D) segments and topology. [Gorte & Pfeifer 2004]

However, this approach lack both efficiency and robustness, which makes it difficult to apply to large trees. A more successful approach was proposed by Xu et al. [Xu *et al.* 2007]. In this approach, an heuristic-based method is first used to reconstruct major tree branches and then small twigs and leaves are synthetized to populate the crown geometry. To do this, points are first connected to their neighbors to form a graph. The local neighborhood of each point is defined by a distance that depend on the scanning resolution. Due to constrained scanning resolution and occlusion, this graph is usually disconnected and has many connected subgraphs as shown by distinct colors in figure 3.12(b). Some reconnection procedures are proposed. Then, the nodes in the graph are segmented into clusters according to the distance to the root. Centers of clusters are then used to generate the main skeleton of the tree. If points are connected in the local neighborhood graph to points belonging to other clusters, then these clusters are connected. A graph of clusters is thus produced and Dijkstra's shortest path algorithms is used to generate an ar-

borescence of nodes. The points in the subgraphs that are not connected to the main skeleton, indicated as green dots in figure 3.12(c), are grouped based on predefined neighborhood. Each group defines a leaf location for placing leaves. Each location is then connected to the main structure. To ensure the self-similarity property of the tree, a pattern in the main structure that form the most similar triangle is copied, scaled, transformed to complete and reach the leaf location.

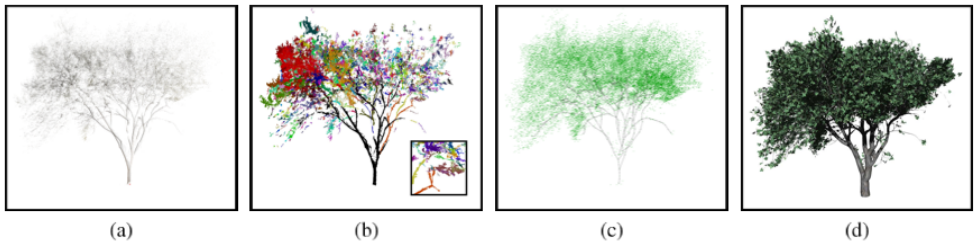


Figure 3.12: (a) A point cloud produced by a single scan of a real tree with the root point marked. (b) A graph is formed by connecting neighboring points. Each connected subgraph is shown here using a distinct color. (c) A skeleton is produced from the graph, and leaf locations (green dots) are identified. (d) The skeleton is extended by synthesizing new small branches growing into the tree crown. A mesh is reconstructed based on the skeleton and leaves are added. [Xu *et al.* 2007]

An extension of Xu method was proposed by Yan et al [Yan *et al.* 2009]. Point segmentation is performed using k-means clustering [Lloyd 1982], which minimizes the energy function defined below.

The input point cloud P is segmented into a set of clusters C. The energy function of a segmentation using k-means clustering is defined such as:

$$E(P,C) = \sum_{i=1}^{n} E(P_i, C_i) = \sum_{i=1}^{n} \sum_{j=1}^{n_i} d(p_j, c_i)^2 \tag{3.3}$$

where c_i is the center of the cluster C_i and $d(p_j, c_i)$ is the Euclidean distance between the point p_j and the center c_i.

For each cluster C_i, a minimal bounding cylinder is computed to measure the tightness of the bounding volume. If the bounding volume satisfies some criteria, the cluster C_i is flagged as fixed and will not change anymore. The remaining clusters that cannot be bounded are subdivided using clustering method again. Finally, the skeletons are transformed into a set of B-spline curves, see figure 3.13.

An alternative method to extract skeletons from scanned point clouds is proposed by Côté et al. [Côté *et al.* 2009]. Points of visible branches are processed using the method of Lazarus and Verroust [Verroust & Lazarus 2000a] and are very similar to Xu et al. [Xu *et al.* 2007]. The k nearest neighbors graph is generated on 3D point cloud. A series of level sets composed of all the points located at the same geodesic distance are extracted. Then, the centers of each level sets is computed. The trunk and some principal branches are generated from these node sets. Finally, they use

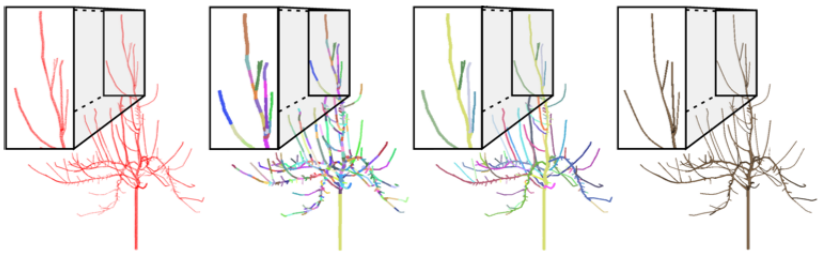


Figure 3.13: From left to right: 1) Scanned data of an apple tree; 2)Segmentation result, different color means different clusters; 3)Branch identification result. The color of each branch is the same as its starting cluster; 4)Final B-spline surface representation of tree branches. [Yan *et al.* 2009]

SCA, described in Section 3.2, to generate procedurally foliage of the structure.

A global optimization method for automatically reconstructing the branching structure of multiple overlapping trees is proposed by Livny et al. [Livny *et al.* 2010]. The algorithm start by connecting each point to its k nearest neighbors and creating a minimum-weight spanning tree over the local neighborhood graph with edge (u, v) of the graph weighted by the Euclidean distance $\|u - v\|^2$ between nodes u and v. Then, a multi-root Dijkstra's algorithm is performed to segment the input points into individual trees. The importance weights, given by the sum of edge lengths in its subtree, are assigned to each vertex of the tree graphs to guide optimization process, as illustrated in Figure 3.14. The edges connecting heavily-weighted vertices indicate long branches. Short branches with low-weight vertices near heavily-weighted samples indicate noise in the dataset. An orientation field is build based on vertices and their importance weights. The orientation of a given vertex will be computed according to the orientation of its parents and to the direction of the edge that connect it to its parent. Low-weight vertices connected to heavily-weighted parents will inherit of the orientation of their parents. Positions of the vertices are updated to reflect their orientations by minimizing partially the difference between the direction of the edge that connect a node to its parent to its computed orientation. The resulting graph exhibits a smooth structure that minimizes distances between original and consolidated sample points. In practice, this process is performed 2 or 3 iterations. Figure 3.14 (right) shows the result of the consolidation step. To reconstruct the tree geometry, a set of generalized cylinders is computed by assign a radius [Tan *et al.* 2007, Xu *et al.* 2007] to each vertex. The vertices are then removed when their adjacent edges share similar orientations and radii. Once the main components of the tree are obtained, fine branches are synthesized using L-system rules from the BSG as in [Tan *et al.* 2007].

Recently, Livny et al. [Livny *et al.* 2011] propose a lobe-based tree representation for modeling trees. Similarly to [Livny *et al.* 2010], a graph first connect neighboring points and construct a shortest-path tree. However, each edge (u, v) is assigned an edge weight $\|u - v\|^\beta$. The manual parameter β is defined according

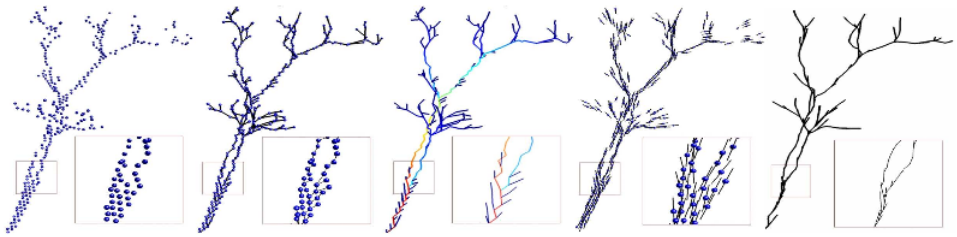


Figure 3.14: The point processing and BSG refinement steps. From left to right: the input points; the initial BSG construction; the importance weights of vertices (shown by edge color); the smooth orientation field; and the smoothed BSG structure

to different species of trees because it allows leveraging the edge lengths to get the main tree structure by using Dijkstra's algorithm. Higher values will assign larger weights on longer edges and thus create a more compact tree structure. The confidence value of each edge is then computed based on tree species. The edges in the graph from the root to the leaves are followed using Dijkstra's algorithm until reach such a low-confidence edge and mark the so far traversed edges as the main branching structure. The remaining edges are now represented as lobes. To compute a lobe-geometry from the points, α-Shape [Edelsbrunner & Mücke 1994, Zhu *et al.* 2008] are used to create convex hulls on points which do not belong to the main branching structure. L-systems are then used to generate a set of branchlets for each species of trees and store them in a library. The final tree is then reconstructed by rebuilding the branching geometry from the skeletal graph and texturing the lobes using predefined elements from the species library. In figure 3.15 illustrates the process of creating and populating lobesfor trees.

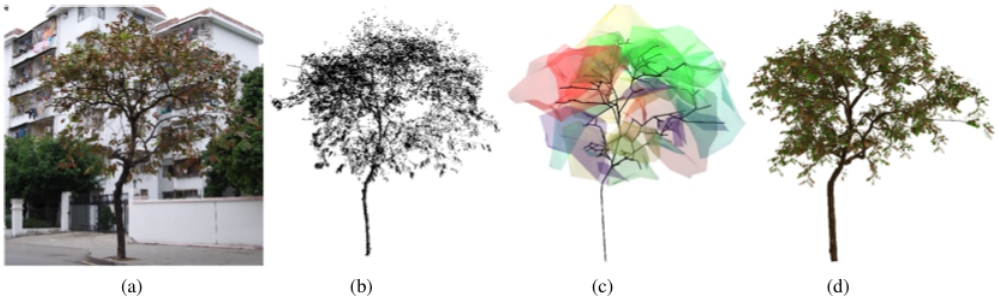


Figure 3.15: Reconstruction of a scanned tree using lobe-based tree representation: a) photograph; b) point set; c) lobe-based representation; d) synthesized tree. [Livny *et al.* 2011]

Conclusion

In this chapter various methods for reconstructing 3D models of plants were reviewed. Techniques based on 3D digitizing, sketching, images, and finally 3D point set were presented. Semi-automatic or automatic techniques proposes impressive results but seems sensitives to occlusion and sparse point density. Plausible structures are produced but without any guarantee of accuracy. In the next chapter, we proposes a new approach that improves the robustness of the reconstruction for noised and sparse data.

Chapter 4

Pipeline of plant modeling from laser scanner

The modeling of plant architecture is commonly developed using computational or simulation techniques. To generate plausible 3D plant structures, there has been a great deal of research on plant modeling using procedural approaches [Prusinkiewicz & Lindenmayer 1990, Deussen & Lintermann 2005]. These techniques, as described in Section 3.2, can create visually appealing plant models. Although realistic complex tree shape can be produced, controlling these processes to achieve a given structure is a challenge. A number of reconstruction methods have been developed that allow for modeling specific plants from real world using 3-D digitizing (described in Section 3.1), or reconstruction based on photographs (described in Section 3.3). With recent advances in laser scanning, precise reconstruction of plants is steadily increasing. These methods, based on point cloud data, tend to create realistic complex plant shape, as mention in Section 3.4.

Our work generalizes techniques for reconstructing real world plants based on point cloud data, obtained from laser scanners. These data are usually incomplete and suffer from significant self-occlusions. We propose a reconstruction pipeline, that can generate models faithful to observed data of plants and improve the robustness of the reconstruction for noised and sparse data.

This chapter is organized as follows: In Section 4.1, we introduce the material that is used to capture geometry of plants. In Section 4.2, we explain how to eliminate artifacts of the point clouds, obtained from laser scanner. In Section 4.3, we gives a first characterization of point clouds from plant scans. In Section 4.4, we give an overview of our pipeline for reconstruction. Our framework for reconstructing structure and geometry of plants is based on three key components, namely *point characterization*(Section 4.5), *skeleton reconstruction*(Section 4.6, 4.7), and *surface reconstruction*(Section 4.8). Finally, we present results produced using our pipeline in Section 4.9.

4.1 Vegetal material

In this work, we use two types of material: laser scanned real plants and virtually scanned plant models. For the first type, we use city trees (Tilia x vulgaris) growing at street alley in Helsinki, Finland scanned with Leica Geosystems HDS2500 laser scanner (see figure 4.2, left) drived using Basis Software Surphaser 25HSX. Figure 4.2 gives original photographs of scanned city trees in Finland. To reduce the occlusion

Figure 4.1: The laser scanners: Leica Geosystems HDS2500 (left) , and Leica Geosystems HDS6200 (right).

problem, the trees were scanned from several positions (from 3 to 4) with a Leica HDS2500. Phased-based ranging technology, described in section 1.2.1, is used for scanning. The scanner specifications give a range accuracy of a single point with 4 mm, and a spot size less than 6 mm in a 0-50 m range of distance [Casula *et al.* 2007].

We also scan a Cherry tree at Clermont Ferrand with Leica Geosystems HDS6200 laser scanner (see figure 4.2,right) of the Centre for Forestry and Climate Change, UK. Similarly, the Leica HDS6200, used the phased-based ranging technology to scan the tree (see figure 4.3). The final scan is also made of multiple scan data acquired from different view angles. The scanner is able to acquire 1 million points per second. It has a single point measurement accuracy of 5 mm at a range of 25 m.

A second type of data is generated from virtual 3D models of trees acquired manually by experts with a magnetic technique. In this study, we used an apple tree [Costes *et al.* 2003]. This model is mainly interesting for validation purpose. Point sets are generated by virtually scanning the 3D models using Z-Buffer capabilities of graphic cards.

Given a surface represented using polygons and a viewing specification, the Z-Buffer algorithm determines visible surfaces. For each pixel of scanning that lies within the boundary of a polygon, the depth z from the camera is calculated. If the z-value of a pixel indicates that the polygon is closer to the camera than the z-value in the current z-buffer, the z-value recorded in the buffer is replaced by the polygon's value. After processing all polygons, the buffer will store the pixels representing visible surface of the object. This algorithm can not only record each pixel position but also can keep the pixel color in the buffer. Figure 4.4 illustrates the point sets of an apple tree generated using z-buffer.

Figure 4.2: An original photography of Lime trees taken in Finland.

Figure 4.3: Original photography of a Cherry tree taken in Clermont Ferrand (France).

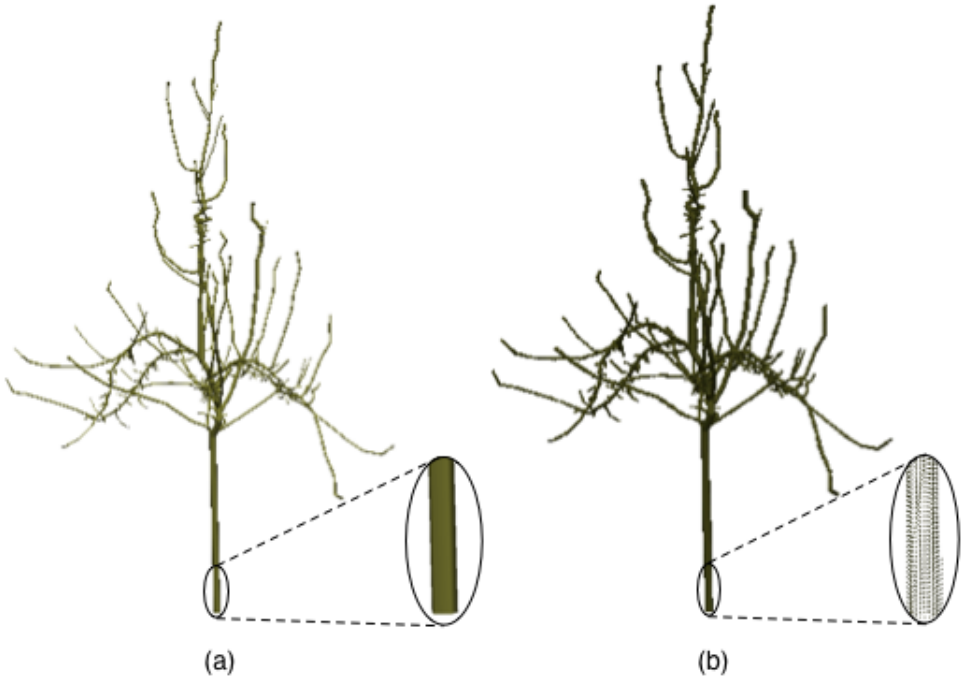


Figure 4.4: 3D model of an apple tree (left), 3D point sets of the model generating by z-buffer algorithm (right).

4.2 Pre-processing

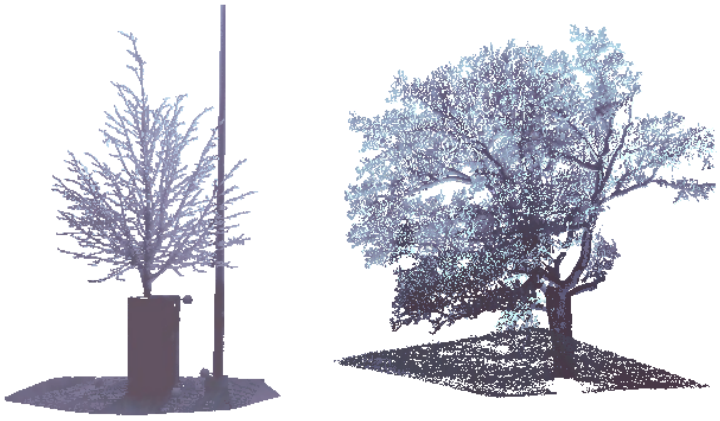


Figure 4.5: Original points data of the plants obtained by Leica Geosystems HDS2500 laser scanner.

Initially, our 3D plant images, obtained from laser scanner in the form of 3D points data, not only consist of the target object but also contain useless objects, such as environmental artifacts (see figure 4.5). We thus need to separate points that belong to the plant from such data.

To eliminate the worthless points, we used the PlantGL library [Boudon 2004, Pradal *et al.* 2009] developed at the Virtual Plants Team. This software is an open-source graphic toolkit for the creation, simulation and analysis of 3D virtual plants. It can typically be used to take a cloud of point data from an ASCII source file and allow the user to select or delete individual points from a graphical display of that data. The PlantGL *Viewer* provides facilities to visualize and process data interactively. Zoom, rotate and other functions are available for moving the object in the viewing plane. Unwanted points can be selected and deleted by using tools through the menu or the toolbar.

Figure 4.7 illustrates the 3D point sets of a Cherry tree after eliminating environmental artifacts. We will use these data as input for reconstructing procedure, described in the next section.

4.3 Points pattern

The distribution of points along surface of a plant mainly depends on the plant architecture and the scanning technology. Therefore, the point cloud of the plants generated from the laser scanners contains varying point density, under-sampling, and noise in the data.

Three types of point patterns can be observed as illustrated in Figure 4.8. Firstly, points are densely distributed on the surface of the trunk and of the main branches. Secondly, a spacer distribution, with some points lying on the surface but others in

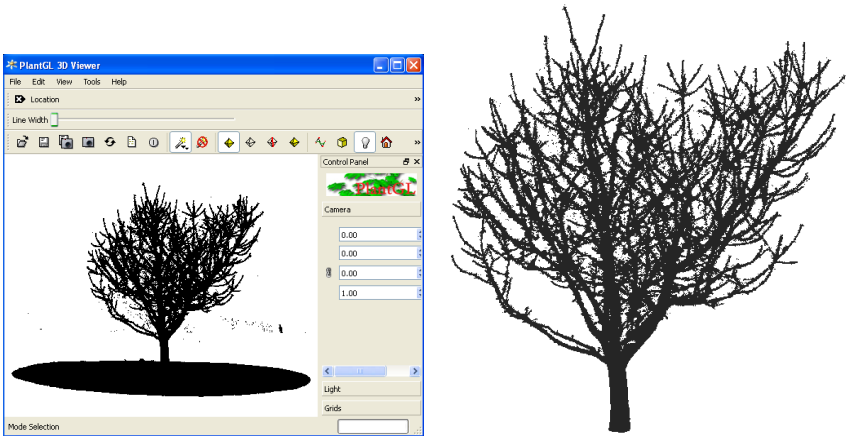


Figure 4.6: Visualization of the cherry tree obtained from the Leica Geosystems HDS 6200 with the PlantGL viewer (left). Cherry tree data points after eliminate environmental artifacts (right).

the volume, due to a larger relative error, can be observed for smaller size branches. Lastly, missing information due to occlusion typically disconnects small branches and twigs into different point clusters. Outliers are also more disturbing, since the error cannot be neglected anymore at the scale of these small branches. This motivates the need for a method robust to noise, and that correctly adapts to the different levels of precisions with which each tree feature is captured.

4.4 Reconstruction pipeline

In this section we outline the general ideas and corresponding processing steps of our algorithm to achieve tree architecture reconstruction. To adapt the reconstruction of trees to the different levels of precision of the laser scans due to the varying scale of tree features, we designed a pipeline based on the following main steps (see Fig.4.9).

- Firstly, the point-set is processed to classify points. In particular, local density relative to the size of the captured feature is estimated and will guide the reconstruction.
- The second step consists in reconstructing the structural skeleton of the tree. This is done through a contraction algorithm that progressively moves points towards the local center of the shape. A point tracking algorithm is then used to extract the skeleton.
- Lastly, reconstruction is achieved by evaluating the radii along the branches of the skeleton, and by computing a 3D model.

The different phases are described in detail in the following sections.

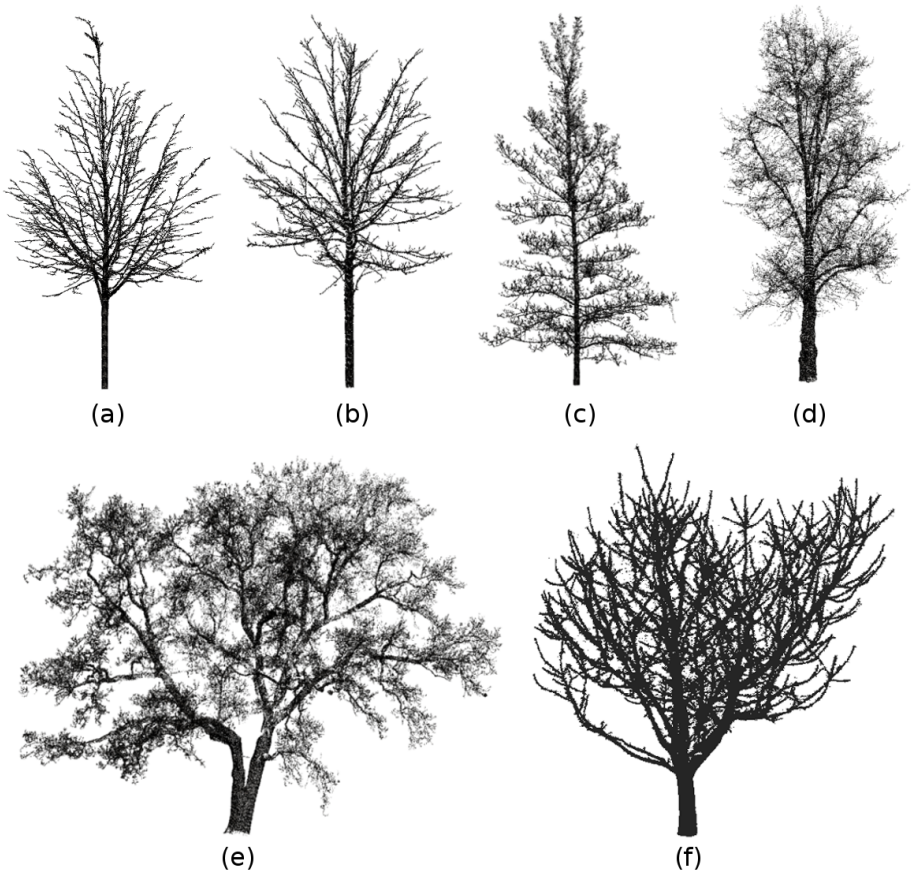


Figure 4.7: Scanned trees with different size: (a) Tree of 5m x 3m with 150k points, (b) Tree of 5m x 3m with 170k points, (c) Tree of 15m x 9m with 180k points, (d) Tree of 19m x 10m with 380k points, (e) Tree of 17m x 21m with 630k points, (f) Tree of 6m x 6m with 1044k points.

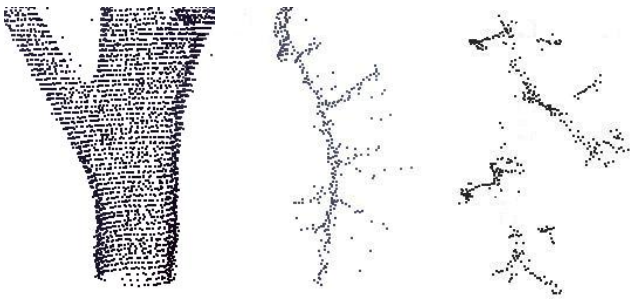


Figure 4.8: Point patterns: first image on left shows dense data points, second image in the middle shows sparse data points, and third image on right shows incomplete data points.

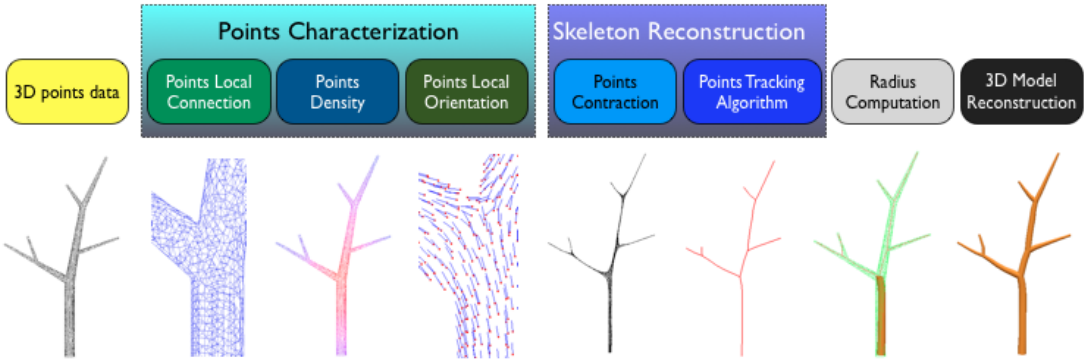


Figure 4.9: The software pipeline for the reconstruction of 3D plant models from laser scans. From left to right, Original data; Point local neighborhood graph; Estimation of the point local density; Estimation of point local orientation; the points contraction scheme; Extraction of the skeleton using point tracking; Estimation of the radii of the branches; Reconstruction of the 3D model

4.5 Points characterization

In this section a number of characterization indices are presented. The main characteristic feature to be computed on the point set is the relative local density, which makes it possible to differentiate main branches from small twigs. We determine and use a neighbourhood of points to associate them such density values.

4.5.1 Local neighborhood graph

By default, point-sets output from scanners are distributed arbitrarily and stored individually without additional information about neighboring points. However, a first crude form of structure can be inferred through the computation of a local neighborhood graph L of order k [Verroust & Lazarus 2000b] called the *Riemannian graph*. This is done by connecting each point to its k closest points in the point-set (we use $k \approx 10$), forming a non-oriented weighted graph where the weight of an edge e_{ij} is the distance between the vertices p_i and p_j it connects. Since such graph can result in several disconnected components, we use a simple reconnection procedure to re-connect it by finding recursively the smallest edge to reconnect two parts.

Let $G_L = (V, E, W)$ denote a weighted graph with vertices-set V, edge-set $E = \{(u, v)|(u, v) \in V \times V, u \neq v\}$, and the edge-weight set W.

Using the neighborhood graph L, the distance between two points in the point-set is redefined as the length of the shortest path in the graph that connects the two points (see Fig. 4.13). Dijkstra's shortest path algorithm [Dijkstra 1959] can be used to compute it. For any point $p \in P$, the shortest distance between the vertices p_i and p_j on graph L can be formulated as:

$$d_L(p_i, p_j) = \sum_{e_k \in path(p_i, p_j)} W(e_k) \tag{4.1}$$

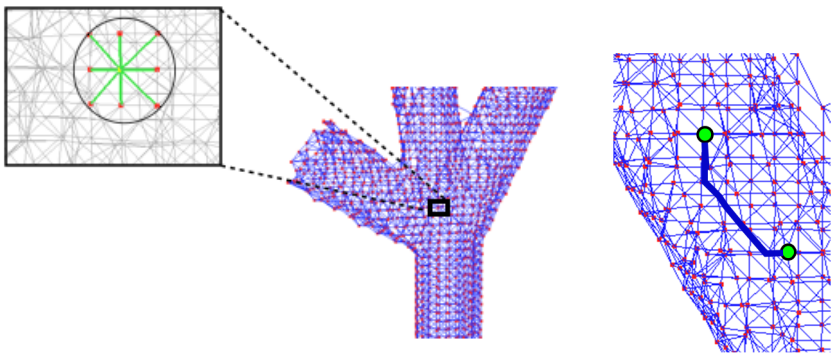


Figure 4.10: Local neighboorhood graph (left), and distance between two points using the shortest path on the graph (right).

4.5.2 Points local density

An important measure for analyzing point cloud is the local density of the points. The aim of points density λ is to describe and characterize the point patterns formed by object that are distributed in three-dimensional space. Pauly et al. [Pauly *et al.* 2002] proposed an estimation formula of the point local density. The local density λ_i at a sample point $p_i \in P$ can be estimated by computing the sphere with minimum radius r centered at p_i that contains the k-nearest neighbors to p_i. They define $\lambda_i = k/r^2$.

However, this limits the estimation of density on small neighborhoods or with large radius different parts of the structure will be mixed. To achieve a more structural criteria, we adapt this method in the following way.

Let λ_i be the local density of a point $p_i \in P$. The k-nearest neighbour is obtained using the shortest path distance d on graph L. A R-neighbourhoud can be determined as all points at a distance R to Then, we define the local density at point p_i as:

$$\lambda_i = \frac{k}{d_L^2} \tag{4.2}$$

Two parameters (k and d_L) affect the estimation of local density of a point. Hence, we analyze in detail the impact of these parameters. First, each local density λ_i is estimated by fixing a distance d_L for all local p_i. In this case, the number of neighbors k for each p_i varies according to the point set distribution. The distribution of local densities with a fixed d_L is represented using histograms in figure 4.11.a. Histograms with 10 and 3 bins are shown. From this distribution, we can calculate an average value of the nearest neighbors ($\bar{k}$). A second set of local densities λ_i are computed by fixing parameter k using $\bar{k}$. In this case, the distances d_L varies for each points. Figure 4.11.b gives the histograms of local densities with k fixed. From Figure 4.11, we can observe that distributions with fixed distance d_L and fixed number of the nearest neighbors k are very similar.

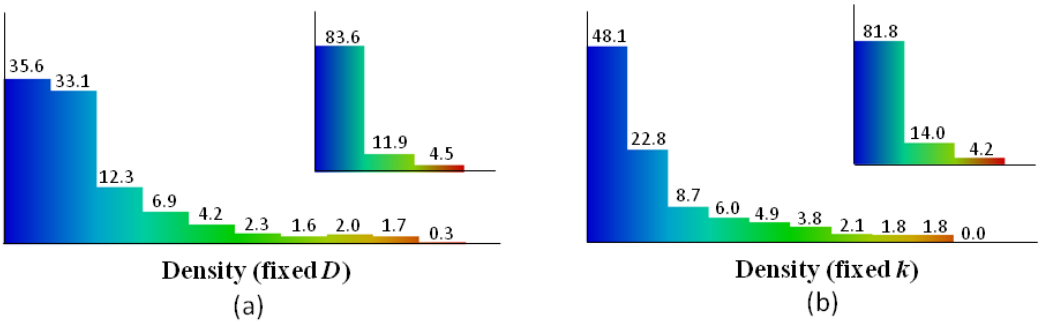


Figure 4.11: Fifteen groups of distribution of the density on a point set: (a) the distance d_L is fixed, (b) the k is fixed.

These histograms help classifying the scanned point set into three point patterns, as described in Section 4.3. Fig. 4.12 illustrates how this local density value discriminates the different regions corresponding to the three point patterns. Thus the different types of branches in a tree can be identified. Red color represents high density and is found preferentially in the central part of the point set, corresponding to the trunk. Green color represents medium density mostly found on the main branches. Blue color represents low density showing the location of small twigs in the structure. This gives an interesting spatial indicator to adaptively tune the parameters of the points contraction procedure, which will be described in section 4.6.

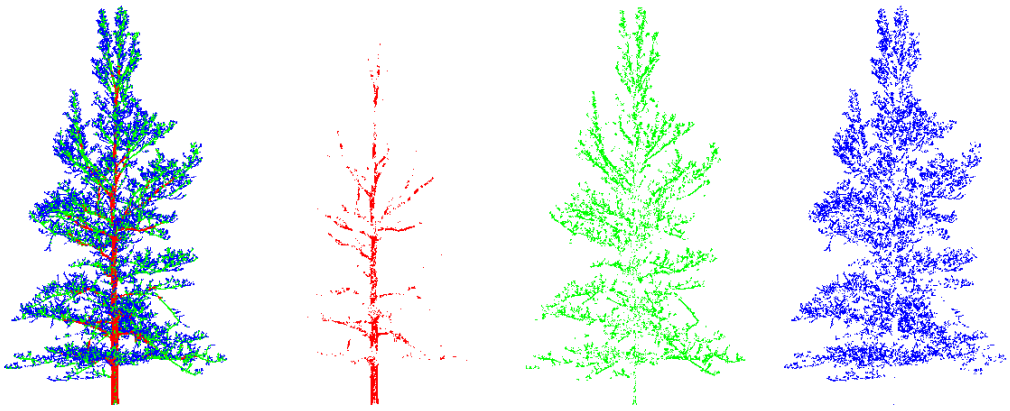


Figure 4.12: Local point density. Red color shows the densest point regions while green color represent the medium and blue color represents the sparsest ones.

4.5.3 Points local orientation

Principal component analysis (PCA) is used in many techniques of point cloud processing to describe neighbourhood's properties through simple statistical analysis. This analysis determine a linear combinations of eigenvectors that form an orthogo-

nal basis for the data set. The significance attributed to each component is related to the amount of variation that the component contributes to the total variation exhibited by the data. From this basis, the significant variables or interactions between variables can be determined, and often the complexity of the data can be reduced to the key significant components [Johnson & Wichern 2002]. In our case, main orientations of a point set is deduced from the eigenvectors.

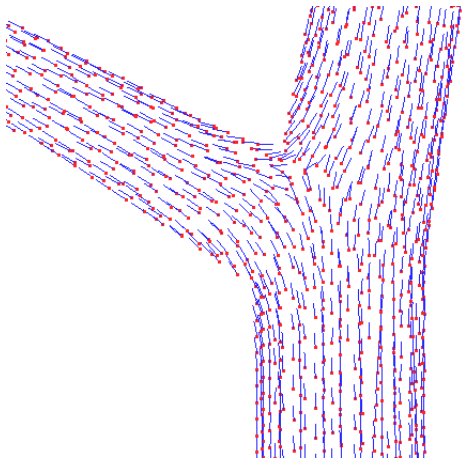


Figure 4.13: Local orientation of points.

Assuming that a point $p \in P$ locally samples the surface of the object to reconstruct, the surface local properties around p can be studied using eigen value analysis of the covariance matrix of the local neighborhood of p [Hoppe *et al.* 1992, Pauly *et al.* 2002]. The covariance matrix Cov for a sample point p is given by

$$Cov = \frac{1}{|N_p|} \sum_{i \in N_p} (p_i - \bar{p})(p_i - \bar{p})^T \tag{4.3}$$

where N_p is the neighborhood of a point p, $\bar{p}$ is the centroid of N_p.

Three orthogonal eigenvectors v_i with $i \in [1, 3]$ corresponding to the principal components of this local neighborhood are computed with the associated eigen values γ_i corresponding to variation of the point set in these directions. With $\gamma_1 \geq \gamma_2 \geq \gamma_3$, v_1 and v_2 give an estimate of the tangent plane that locally approximates the surface near p_i, while v_3 represents the normal to the local surface [Hoppe *et al.* 1992, Pauly *et al.* 2002].

In this approach, we focus on the main orientation of the point-set, by associating the value v_1 with the point p_i, v_1 being the main direction of local variation of the local neighborhood around p_i (see Fig. 4.13). For a cylindrical shape, this direction corresponds to the axis of the cylinder which we aim at reconstructing. Note that the orientation of the main direction is not consistent over the point set as two close points may be given similar, but opposite direction. To assess consistency to the orientations, we use a similar reorientation procedure that the one originally proposed for normal vectors in [Hoppe *et al.* 1992], see Section 1.3.2.

4.6 Points contraction

A useful structure that represents the geometry and topology of a 3D object is curve skeleton. A curve skeleton is useful in practice due to its topological simplicity, leading to computational efficiency and ease of manipulation. Many algorithms for curve skeleton extraction exist for standart object, as described in Section 1.5.2.1. However, contracting points of scanned trees raises a number of issues, which prevents us from using standard methods. In particular, sparse data sets, outliers, and measurement errors on the silhouettes of branches make it difficult to robustly compute the normal or laplacian information typically required [Tagliasacchi *et al.* 2009, Cao *et al.* 2010]. To avoid this, we make use of density and orientation values to parameterize an alternative method inspired from the work of Giannitrapani and Murino [Giannitrapani & Murino 1999].

Giannitrapani and Murino proposed a method to contract point sets distributed on the surface of an object. In their work, they used data acquired by acoustic camera sensing, but we generalize the method to data coming from a laser scanner. In this algorithm, each point is shifted from the border of the shape towards its center by determining for each point a neighborhood and move the point to the center of the neighborhood. This can be seen as a mesh Laplacian operator [Tagliasacchi *et al.* 2009]. However, when testing the algorithm with both simple and complex structures, we found that some distortion occurs, especially at the branching points (see fig 4.14), when using large neighborhood or only too small contraction can be achieved when small neighborhood are used. To alleviate this problem, we propose several adaptations. First, we define to use an anisotropic distance (d_L) based on the Riemannian graph, described in 4.5.1, to determine point neighborhood to contract. Second, an adaptive distance is used and defined according to the local density information to determine the neighborhood of each point.

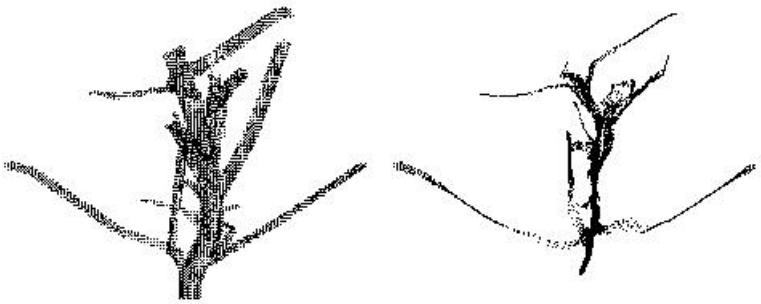


Figure 4.14: Illustrated distortion of object. Original data (left), Contraction data (right).

An ideal neighborhood for computing the contraction to apply on a point p_i would include an entire section of the branch it belongs to. To achieve this, we use an anisotropic distance that extends neighborhood preferentially in the radial direction along branches, see figure 4.15. To define such an anisotropic distance

d_t, let's consider a vector $\vec{v}$ representing for instance the displacement between two points. $\vec{v}$ can be decomposed into its components in the $\vec{t}$ direction and its radial component perpendicular to $\vec{t}$. In this case, $\vec{v} = a\vec{t} + \vec{b}$ with $a = \vec{v}.\vec{t}$ and thus $\vec{b} = \vec{v} - a\vec{t}$. We then define the anisotropic distance as:

$$d_t(\vec{v}) = \sqrt{\alpha * a^2 + \beta * \left\|\vec{b}\right\|^2} \tag{4.4}$$

We choose α equal to 1 and β to $1/\pi$. With such values, a circle of radius 1 in a plane perpendicular to $\vec{t}$ will be of circumference length 2 and thus two opposite points on the circle will be at a distance 1. For each point p_i, the t direction is defined as the local orientation t_i computed from PCA as explained in previous section. To compute the contraction neighborhood of a point, we first redefine the weight of the edges of the graph of local connection as $e_{ij} = d_{t_i}(p_i - p_j)$. We then select points which are at a distance inferior to the radius of contraction R in the graph L.

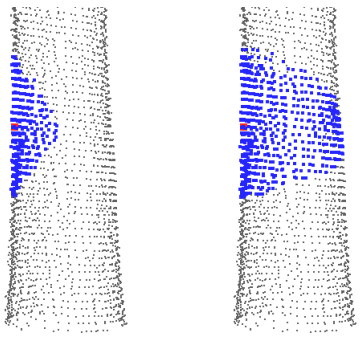


Figure 4.15: Local neighborhood approximation for computing the contraction. using the euclidean distance (left), using an anisotropic distance.

Due to the different size of elements of a tree (as described in section 4.3), the radius of contraction R should be adaptive. We propose to correlate this radius to the local density. Here, we assume a relationship between the precision of the description and the size of the element described. Indeed large branches produce a dense sampling while small branches are usually sparsely described as shown by the density characterization in section 4.5.2. A function R_λ is defined on the range of densities $[\lambda_{min}, \lambda_{max}]$ to associate a radius of contraction to each λ, see figure 4.16. To build such functions, radii of contraction are defined as the average radii of the branches represented with a given density. Samples of branch radii and are measured manually on the scanned data and correlated with corresponding densities. Interpolation between the measured samples is used to fully define the function.

Given a point p_i and a radius r_i, we next compute the corresponding contraction point c_i, the centroid of the neighborhood. The neighborhood is noted S_i. The centroid c_i of S_i is defined in the following way:

$$c_i = \frac{\sum_{p_j \in S_i} p_j}{|S_i|} \tag{4.5}$$

Each point p_i of the point set P is finally contracted by shifting it from its position toward the centroid C_i of its neighborhood.

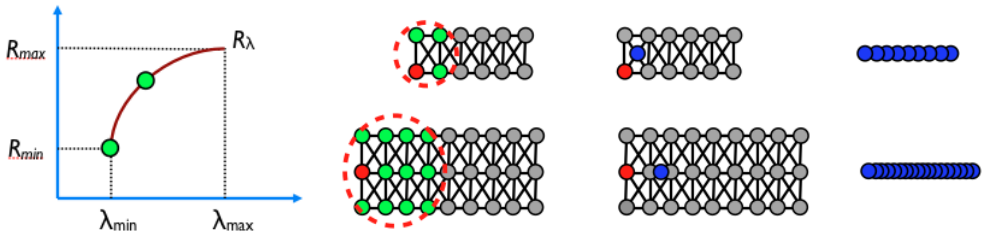


Figure 4.16: Radius of contraction $r_i \in R$ is determined according to the local density and user defined function. If the points have low density, the radius of contraction is small. On the other hand, if the points have high density, the radius used for contraction is big. Points set belonging to trunk or big branches will be more contracted than small branches or twigs

4.7 Skeleton reconstruction

After transforming the original points set into a more compact form, we then extract the skeleton of the shape representing the architecture of the tree. Our work is inspired from the space colonization algorithm (SCA) proposed by Runions et al. [Runions *et al.* 2007] and Palubicki et al. [Palubicki *et al.* 2009], for generating artificial branching structures. As introduced in section 3.2, the key idea of SCA is an iterative addition of new elements (nodes) to the tree structure using competition for space as the driving force of pattern formation. Points used to parameterized SCA were usually sparsely distributed in the desired volume of the plant. We adapt it to point sets coming from laser scans to retrieve faithfully file of points describing linear segments of the plants. In particular we extend it by making it adaptive to the different levels of precision of the point set.

4.7.1 The point tracking algorithm

The operation of the algorithm begin with an initial contracted point set, called *attractors*, $P = \{p_i\}_{i\in[1,n]}$. The branching structure that consist of skeleton nodes $T = \{t_j\}_{j\in[1,m]}$ is generated iteratively. During each iteration, attractors influence the nearest skeleton node t_i. This influence occurs if the distance between the attractors and the closest skeleton node is inferior to the *influence distance* r_p. If the attractors set is not empty, new skeleton nodes will be created at a *growth distance* r_g from t_i and attached to t_i in the direction of the influencing attractors

as described below. The neighbourhoods of each new node t_i within a threshold *kill distance* ρ are then deleted. This process terminates when all attractors have been removed, or when no nodes are within the influence distance of the remaining attractors.

We start from an initial skeleton node at the base of the trunk. At each iteration, the distances (influence, kill and growth distance) are made sensitive to the point local density of t_i. For dense region, the algorithm is more reliable with big distances to avoid small surface variations, while short steps are recommended for sparse regions to detect small twigs. An expert function defines a mapping between the adaptive distances and local density, see figure 4.17. Adapting distances make it possible to follow more closely the different size of elements in the structure.

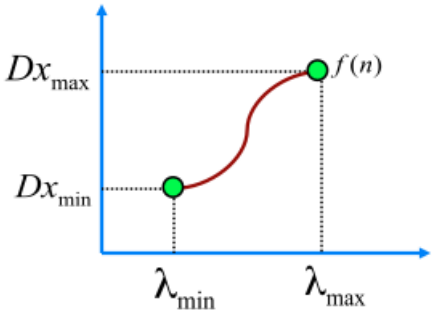


Figure 4.17: The adaptive distances (influence, kill, and growth distance) are determine according to the local density and user defined function.

At each iteration, new nodes can be generated from a parent node in several directions to model branching points. The set of attractors $S(t_i)$ within a distance r_p are thus classified into groups using conical perception volumes or a clustering method. Indeed, in case of dense region, we classify the attractors from a statistical analysis of the local point set around the node t_i, (see figure 4.18.b). This is achieved by clustering the local orientation of $S(t_i)$, as described in Section 4.7.2. Sparse regions are more difficult to cluster since not enough information is available to establish significant statistical analysis. In this case, conical volumes of perception are used. They are defined with a angle θ and a distance r_p to sense the surrounding space, (see (figure 4.18.a)).

Let a group of attractors denoted by $s(t_i)$. A new skeleton node t_{i+1} is positioned at distance r_g from t_i, in the direction defined as the average of the normalized vectors toward all the attractors $p \in s(t_i)$. Thus, $t_{i+1} = t_i + r_p\hat{u}$, where

$$\hat{u} = \frac{\vec{u}}{\|\vec{u}\|} \quad \text{and} \quad \vec{u} = \sum_{p \in s(t_i)} \frac{p - t_i}{\|p - t_i\|} \tag{4.6}$$

When a new node is created, the position is shifted to the position of the closest attractor to make sure that the skeleton follows the point-set pattern, see figure 4.19. A new node is attached by default to its parent node t_i, see figure 4.20.a.

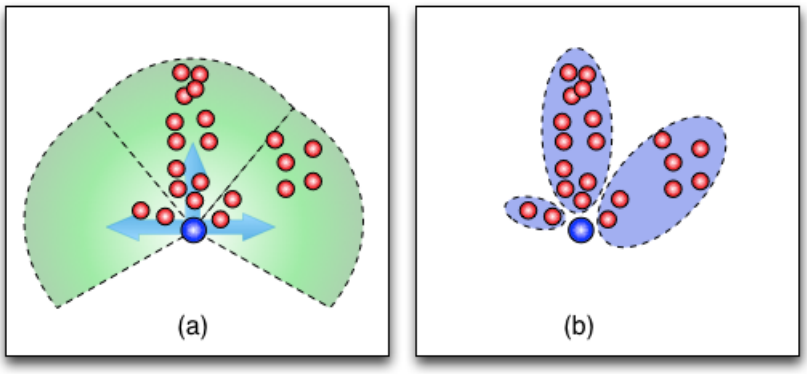


Figure 4.18: Two methods for classifing attractors: (a) using conical perception volume, (b) using a clustering method.

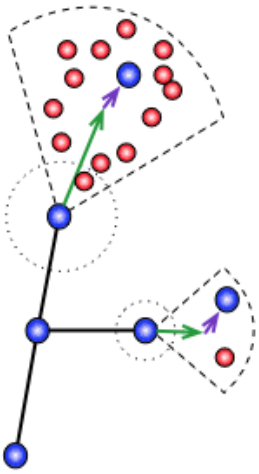


Figure 4.19: When the new nodes are generated, it will be shifted to the position of the closest attractor. This ensures that the algorithm follows the pattern given by the points.

In case of large branches, it is still possible to generate too artificial lateral nodes due to the width of the branch. To prevent this problem, we specify an additional procedure to validate creation of lateral branches. Let an initial growth direction of t_i computed from previous node $\overrightarrow{t_{i-1}t_i}$. We first classify the child nodes of t_i as axial and lateral nodes. The nodes whose direction $\overrightarrow{t_it_{i+1}}$ is the closest to the parent direction of t_i is supposed to be axial and the other laterals. We then test for each lateral nodes if the angle between its direction and the direction of the axial node is smaller than a threshold. If yes, the lateral node is deleted.

If a lateral node is preserved, the algorithm will then refine its connection. Let a, b new nodes connected to the t_i. The segment (t_i, a) is supposed to be the axial branch and (t_i, b) the lateral branch. Let consider the direction $\vec{v}$ defined as the average local orientation of the attractors $s(t_i)$ responsible of the generation of the node b. The local direction $\overrightarrow{v}$ is prolongated up to the segment (t_i, a). At the intersection x, a new parent node of b is created.

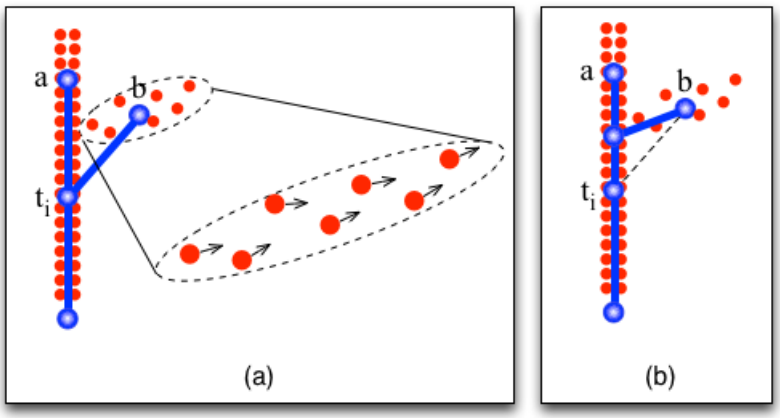


Figure 4.20: Connection refinement on a skeleton using local direction of the point set at the position of the new node.

Then, the neighbourhoods of new nodes within a circle radius (kill distance) ρ are removed and the process continues until no attractors are left or can influence any nodes.

4.7.2 Determining the branch directions by clustering point orientations

To cluster attractors $S(t_i)$ of a node t_i, the node has an associated orientation frame F_i with heading H, up U and left L directions. Heading direction is given by the growth direction and the two other ones are deduced from the orientations of parent nodes. The orientation vectors of all attractors around t_i are re-expressed in the local frame F_i, as polar coordinates $v_j = (\theta_j, \varphi_j)$. To determine the number of branches at this point, we perform a clustering on the polar coordinates of the orientations of the attractors.

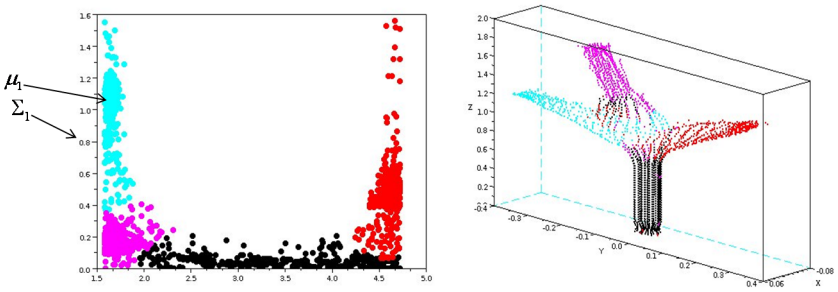


Figure 4.21: Clustering with gaussian mixtures

Clustering of these polar orientation values using bivariate Gaussian mixture models gives us local main directions of the clouds and thus the possible branch directions. Gaussian mixture models have been used extensively for cluster analysis [Biernacki *et al.* 2001]. Clustering relies on the following model for the probability density function (PDF) of v_j:

$$f(\theta_j, \varphi_j) = \sum_{c=1}^{C} P(H_j = c) f_{N(\mu_c, \sum_c)}(\theta_j, \varphi_j),$$

where $H_j \in \{1, \ldots, C\}$ is a hidden qualitative random variable that represents the cluster of v_j, $f_{N(\mu_c, \Sigma_c)}$ is the Gaussian PDF of the conditional distribution of v_j given its cluster $H_j = c$ (with mean μ_c and covariance matrix Σ_c), and the $P(H_j = c) = \pi_c$ are positive weights with $\sum_c \pi_c = 1$. An assumption is made that the elliptical shaped clusters have a Gaussian distribution.

The MIXMOD software [Biernacki *et al.* 2006] is used to estimate the parameters $\xi = (\pi_c, \mu_c, \Sigma_c)_{1 \leq c \leq C}$ by maximization of the log likelihood. The clustering step is achieved by estimating H_j using its most probable value $\hat{h}_j$ given v_j. The number of clusters C is estimated using the Integrated Classification Likelihood (ICL) of Biernacki *et al.* [Biernacki *et al.* 2001]. ICL is defined as

$$\mathrm{ICL}(C) = \log f_{\hat{\xi}}((v_1, \ldots, v_{l_i}), (\hat{h}_1, \ldots, \hat{h}_{l_i})) - \frac{d_C}{2} \log(l_i)$$

where l_i is the number of nodes in $S(n_i)$, d_C is the dimension of ξ and $\hat{\xi}$ is the maximum likelihood estimator of ξ. ICL is a compromise between model fit and separation between clusters on one hand (assessed by $\log f_{\hat{\xi}}$) and the model complexity on the other hand (assessed by $d_C \log(l_i)$). It thus reflects the number of degrees of freedom in the model.

4.8 Surface reconstruction

The last step of the process is aimed at modeling the surface of the plants. The algorithm operates in two passes. First, a radius at each skeleton node is estimated. Second, the surface representation of tree model is computed.

4.8.1 Estimation of the diameter

Figure 4.22: Diameter estimation of a scanned tree. Original point data (left), and computer generated tree model (right).

A radius at each skeleton node is estimated using the pipe model [Shinozaki *et al.* 1964]. In this model, the diameter of a node n_i which carries two children nodes n_j and n_k are linked with the relation $d_i^p = d_j^p + d_k^p$ with p being the pipe exponent parameter.

The value of p is computed from the data as follows: The relationship above being recursive if n_j and n_k also carry children nodes, we derive these equations until reaching the leaves n_{f_i} of the subtree rooted in n_i and establish that:

$$d_i^p = \sum d_{f_i}^p. \tag{4.7}$$

Let us now consider n_i to be the root node of the reconstructed tree. Its diameter d_i can easily be computed from the scan by fitting a circle [Welzl 1991] to a section of the scan at its basis. We found that diameters at branch endings are difficult to estimate. Thus we fix them to a small value d_{f_i} for all the f nodes n_{f_i}. As a result, value of p can be estimated as:

$$p = \frac{\log f}{\log d_i - \log d_{f_i}} \tag{4.8}$$

Figure 4.22 illustrates 3D model a tree when diameter is calculated.

4.8.2 Reconstruction of the surface

To define the surface of the branches, we use a generalized cylinder proposed by Bloomenthal [Bloomenthal 1985]. This geometric model is useful for modeling tubular shape. Given a 2D cross-sectional curve and a 3D skeleton curve, the generalized cylinder is defined as the sweep surface of the cross-sectional curve moving along the skeleton curve. The cross-sectional plane can be constrainted to be orthogonal to the tangent direction of the skeleton curve. To polygonize the surface, a finite number of cross sections are evaluated along the skeleton curve and connected together, see figure 4.23.

Figure 4.23: Generalized cylinder with cross-sectional circles.

Reconstructing the surface of our model is defined from the skeleton curves and cross section curve. The circular cross sections of varying radii are calculated by estimated the diameter of the skeleton nodes. A original point set and resulting reconstructed branching structure are compared in the close view in figure 4.24.

Figure 4.24: Comparison between a close view of a reconstructed complex branching structure (right), and the same view of the original point set (left).

4.9 Results of the reconstruction

The results of different steps of the reconstruction process are illustrated on Fig. 4.25 on two exemplars of trees. It consist of original point data, contracted data,

extraction of the skeleton with the point tracking algorithm, and reconstruction of the surface.

Figure 4.25: The different steps of the reconstruction process on two city trees. From left to right: original point data, contracted points, extraction of the skeleton with the point tracking algorithm, and reconstruction of the surface.

More complex examples are illustrated on Fig. 4.26. We also model the trees with leaves. To achieve this, we use L-Py [Boudon *et al.* 2012], a simulation package that mixes L-systems construction with the Python high-level modeling language, for placing leaves on along branches of the trees.

The quality of the reconstruction can be visually assessed in Fig. 4.27 with a side by side comparison with a photograph of the original tree. 3D reconstruction of the Cherry tree is rendered and integrated on the same background of the original photograph. A more quantitative validation method is presented in next chapter.

Finally, a large scene with different species of the trees is rendered to test the large-scale of forest ecosystems, see figure 4.28.

Figure 4.26: Different results of reconstruction on trees of different sizes. Top left: City tree of 4m high x 3m large scanned with 150k points. Top right: Tree of 19m x 10m with 380k points. Bottom: Tree of 17m x 21m with 630k points. Foliage is added procedurally.

Figure 4.27: Reconstruction of a cherry tree. Left: photograph of the original tree. Right: 3D reconstruction from a laser scan rendered and integrated on the same background.

Figure 4.28: A large scene with different species of the trees.

Conclusion

This chapter has presented a robust method for reconstructing complex trees from laser scans, despite of the locally sparse data. The reconstruction method relies on the characterization of the point set to determine local levels of precision and local orientations of the point set. An adaptive contraction procedure enables us to concentrate points near the center of the branches, and a point tracking procedure is then used to extract the skeleton of the tree. This last step builds on a method for simulating tree growth, that we modified to achieve the generation of an actual tree.

Chapter 5

Evaluating the reconstructed model

While we have focused on the reconstruction of branching structures of scanned plants in the previous chapter, we will discuss the evaluation of this reconstruction in this chapter. The quantitative validation of a reconstructed architecture compared to a real plant is an important task. Some previous works can produce realistic looking trees, however they do not propose any validation and therefore give no guarantee concerning the usefulness of their model in a biological context. To address this problem, we investigate evaluation procedures.

We first summarize related work on plant similarity evaluation based on global or geometrical (Section 5.1) and structural consideration (Section 5.2). In Section 5.3, we describe more in details the structural comparison method proposed by Ferraro and Godin [Ferraro & Godin 2000] between plant architectures. In Section 5.4, we adapt and apply this method to our laser reconstruction to evaluate their accuracy. In this case, an automatic reconstruction is compared to expert reconstruction. In this first approach, plant topological structures weighted with geometrical attributes are compared. However, we found that geometrically similar elements can be dissimilar because of particular topological difference in the structure. In Section 5.5, we propose an alternative algorithm that will first evaluate geometrical similarity between element and use it to guide the evaluation of topological similarity between the 2 structures.

5.1 Global comparison methods

A first approach for comparing similarity between plant models consists of summarizing each individual by a small number of global variables such as wood content, foliage area, crown volume, etc. The similarity of the models is then reduced to the proximity between these synthetic variables. We present such methods used for evaluating reconstruction in this section.

5.1.1 Architectural plant properties evaluation

The evaluate the accuracy of its reconstruction method, Côté et al [Côté *et al.* 2009] propose to compare scanned 3D point clouds used as reference and a reconstructed model. In their study, a 3D point cloud was generated from a first reconstructed model by using either physically-based ray tracing software (see

[Pharr & Humphreys 2004]) or Z-buffering (described in section 4.1). The 3D point cloud is encompassed into a grid of voxels. This makes it possible to compare density of element in each voxel.

Additionally, they propose to compare geometrical properties of a reconstructed and the corresponding reference tree models. For instance, the volumes of the convex hulls or the total leaf and wood area are compared. These areas are calculated by summing each object's total surface area of foliage and wood component respectively. To express the difference, the following formula is used. Let N be the number of elements, x_i is the value of parameter of interest for the reconstructed model, and y_i is for the reference model. Differences are expressed in percent and are computed as:

$$\Delta = \frac{100}{N} \sum_{i=1}^{N} \frac{x_i - y_i}{y_i} \tag{5.1}$$

The root mean squared deviation (RMSD) can also be reported as an indicator of agreement between the reference and the reconstructed tree. The RMSD was computered as:

$$RMSD = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - y_i)^2} \tag{5.2}$$

5.1.2 Radiative canopy properties evaluation

The evaluation is performed under identical illumination conditions by simulating reflectance and transmission of light by the virtual plant models. The resulting differences can be directly attributed to the structural differences between the reference and reconstructed models [Côté *et al.* 2009].

Canopy reflectance signatures comparison This evaluation method can be taken by ingesting a reference and a reconstructed model into a validated radiative transfer (RT) model that simulates their reflectance properties. The 3D Monte Carlo ray-tracing model [Widlowski *et al.* 2006] is used to simulate reflectance factors. The difference of reference and reconstucted plant canopies are conditioned by effective interception of solar radiation that provides the necessary energy for photosynthesis and other physiological processes.

Hemisperical photographs comparison This evaluation method estimate solar radiation and characterize plant canopy geometry using photographs taken above the tree and looking upward through an extreme wide-angle lens. Hemispherical views are particularly interesting since they can directly calculate solar radiation transmitted through (or intercepted by) plant canopy. Figure 5.1 illustrates this method. A hemispheric picture is taken from a real plant at a given location, while an equivalent virtual picture is computed with the same camera position on the reconstructed plant. The pictures are segmented, which means that every black and white pixels corresponds to obstruction by foliage, wood or gap in the canopy respectively. The amount of intercepted light is summarized with the *canopy openness index* defined as the ratio between white pixels and total number of pixels on the picture

[Pradal *et al.* 2009]. The canopy openness index is used to evaluate similarity of distribution and density of the foliage of a tree.

Figure 5.1: Evaluation of plant model reconstruction using hemispheric view [Parveaud 2006]. On the left, an hemispheric photograph of a real walnut from the ground. On the right, reconstructed mockup using PlantGL ([Pradal *et al.* 2009]) exported to Pov-Ray [POV-RAY].

5.2 Structural comparison methods

In contrast to Section 5.1, domains exist in which plant topological structure plays an important role [Ferraro & Godin 2000]. This approach, based on a piece-by-piece comparison of plants, leads researches to consider more detailed descriptions of plant crowns.

In the early seventies, Wagner and Fischer [Wagner & Fischer 1974] presented an algorithm, string-to-string correction, for computing the distance between two strings of characters as the minimum cost sequence of elementary operations needed to transform one string into the other. The operations they consider are: (1) changing one character to another single character, (2) deleting one character from the given string, (3) inserting a single character into the given string. The various attempts to achieve high-dimensional generalizations of string have been proposed by [Selkow 1977, Tai 1979, Lu 1979]. These works, tree-to-tree correction, consist in computing a distance between two tree-graphs as the minimum cost of a sequence of elementary operations that converts one tree-graph into the other. The core structure of the algorithm is based on the dynamic programming principle and determines an optimal mapping between tree-graphs. Since plant architecture representations are unordered, as mentioned in Section 2.2.2, these algorithms cannot be used directly to the problem of plant comparison. However, Zhang [Zhang 1993, Zhang 1996] proposed an algorithm in theoretical computer science for computing a distance between unordered rooted tree graphs. Ferraro and Godin [Ferraro & Godin 2000] adapt it for plant architectures comparison. This algorithm

will be the basis of a first part of our work and thus will be described in details in the next section.

5.3 A distance measure between plant architecture

The computation of an edit distance between two tree-graphs consists in determining a sequence of edit operations of minimum cost which transforms an initial tree into a target tree [Ferraro & Godin 2000]. An optimal sequence of edit operations (i.e. a sequence having a minimum cost) from T_1 to T_2 is such that the corresponding mapping associates a vertex of T_1 to at most one vertex of T_2 and reciprocally.

5.3.1 Edit operations

Following Wagner and Fisher [Wagner & Fischer 1974] original definitions on sequence, three edit operations are considered: *substituting* a vertex x with a vertex y means changing the label of x into the label of y, *deleting* a vertex x means making the children of x become a new children of the father of x and then removing x, *inserting* a vertex y means that y becomes the child of a vertex z and a subset of consecutive children (relatively to their order) of z becomes the set of children of y, see an example in figure 5.2.

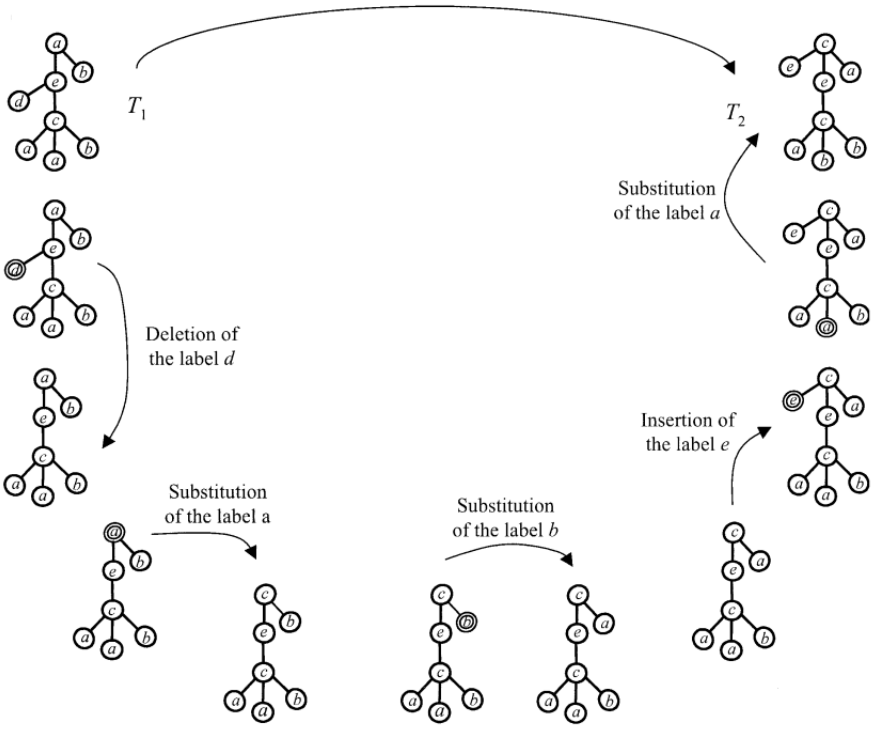


Figure 5.2: The five edit operations used to transform T_1 into T_2. [Ferraro & Godin 2000]

Let s be an edit opertion and d be a distance, a cost γ is assigned to each edit operation as follows:

- $\gamma(s) = d(x, y)$ if s substitutes x into y,
- $\gamma(s) = d_{del}(x) = d(x, \lambda)$ if s deletes x,
- $\gamma(s) = d_{ins}(y) = d(\lambda, y)$ if s inserts y.

The cost γ is extended to a sequence of edit operation $S = \{s_1, s_2, \ldots, s_n\}$ by letting:

$$\gamma(S) = \sum_{i=1}^{n} \gamma(s_i) \tag{5.3}$$

The set of possible edit operation sequence which transform T_1 into T_2 is denoted by $S(T_1, T_2)$. The dissimilarity measure $D(T_1, T_2)$ is then defined as the minimum cost of a sequence in $S(T_1, T_2)$:

$$D(T_1, T_2) = \min_{S \in S(T_1, T_2)} \{\gamma(S)\} \tag{5.4}$$

5.3.2 Edit mappings

An *edit distance mapping* (or just a *mapping*) between T_1 and T_2 is a representation of the edit operations. Figure 5.3 illustrates a mapping that corresponds to the edit operation in Figure 5.2. Formally, the triple (M, T_1, T_2) is an edit distance mapping from T_1 to T_2, if M is a set of ordered pairs (x, y) of vertices from $T_1 \times T_2$ (x and y are images of one another). Let N_1 and N_2 be the set of vertices in T_1 and T_2 respectively not in a pair of M. The set of all possible mapping from T_1 to T_2 is denoted by $\mathcal{M}$. The cost of $M \in \mathcal{M}$ is given by:

$$\gamma(M) = \sum_{(x,y) \in M} d(x, y) + \sum_{x \in N_1} d(x, \lambda) + \sum_{y \in N_2} d(\lambda, y) \tag{5.5}$$

The minimum cost mapping can then be equivalent to the edit distance:

$$D(T_1, T_2) = \min_{M \in \mathcal{M}} \{\gamma(M)\} \tag{5.6}$$

To provide meaningful distance, the cost of the edit operations should be parametrized according to the targeted application. We choose to parametrized them to reflect the geometric dissimilarity between individual segments. This will be described in the next section.

5.4 Comparison of plant architectures

Rigorous evaluation of the plant reconstruction procedure would require to access to an actual structure of the plant. Unfortunately, no structural elements are obtained from the scanning in addition to the point cloud data. We overcame this limitation

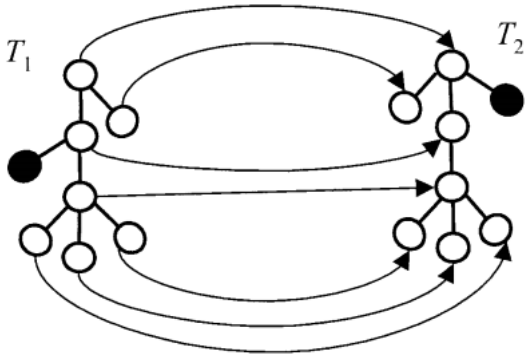


Figure 5.3: The mapping corresponding to the edit operation in Figure 5.2. [Ferraro & Godin 2000]

by getting actual structure of plants from two types of material. The structure of an apple tree was acquired manually by expert using magnetic devices (see Section 4.1). This provide a reference structure and we virtually scan it to have point data set on which we can test our reconstruction procedure.

For each plant taken from the laser scanners, a reference 3D model was built by editing the skeleton resulting from automatic tree reconstruction. To do this, the user or biological expert can use a visual tool that we developed to assess relevance and accuracy of models in their biological context. The visual tool, that we made, can edit the skeletal structure of each tree by adding, deleting, repositioning or reorganizing segments in the structure according to the original point set. The edited skeletal structure was then considered as a reference structure for further assessment of the pipeline tests on each of plant.

As a result, both reference and reconstructed structures are represented by a multiscale tree graph (MTG), described in Section 2.2.2, whose nodes are associated with branch segments. However, the scale used in both representations may be different (user may use for instance longer or shorter segment elements) which makes the direct comparison of the structure difficult.

5.4.1 Homogeneizing skeletal structures

Before comparing a test reconstruction produced by the pipeline against the reference reconstruction, some homogeneization procedure had to be carried out. Indeed, the two compared treelike skeletal structures are in general composed of different types of segments with different sizes. Therefore, we considered the branch scale as useful for comparing the models. However, we found that the beginning and the ending of the branches are sometimes difficult to determine automatically. In some architectures, it requires careful study of the local geometry and topology to identify botanical markers such as scars. These markers are difficult to identify with laser scanner data. In particular with sympodial structures at a branching point,

we cannot distinguish between the continuation of a main branch and an emerging child branch.

The problem can be avoided by considering scale of segments between arbitrary branching points, see figure 5.4. We homogeneized both skeletal structures by re-segmenting both trees in terms of *inter-ramification branch segment* (IBS). IBS are one-piece segments connecting two branching points in a skeletal tree-like structure. At this scale, new components will be created if any parent node have two or more children. We use this scale to perform a structural comparison as explain in the next section.

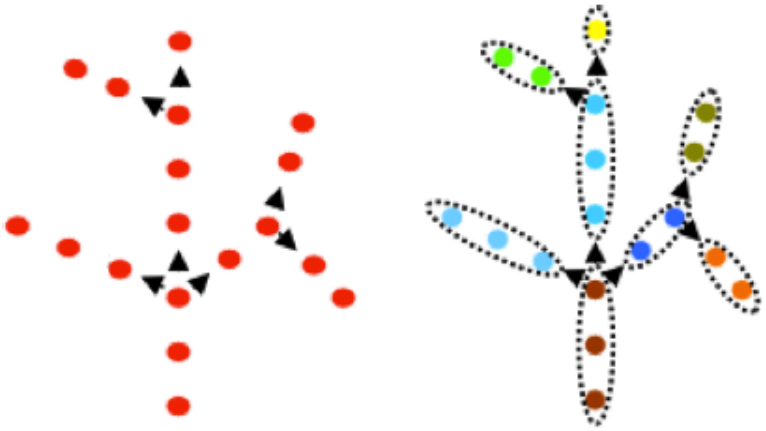


Figure 5.4: Homogeneized skeletal structures

5.4.2 The local cost function

Based on the homogeneized structure, the comparison of the reconstructed and reference skeletons could then be carried out. We evaluate both geometrical and topological similarity between them. This is done by calculating the minimum cost of edit operations, described in Section 5.3, to transform first tree into second one with cost functions weighted by geometrical comparison between individual nodes.

As mentioned above, it is necessary to consider an elementary distance between the components of the sequences or tree graphs. In our case of plant comparison, a local cost function assigns to each pair of entities (x, y) of two plants T_1 and T_2, a non-negative real number, called a cost, for deleting x, for inserting y, and substituting x into y.

For any two IBSs that can possibly be associated in T_1 and T_2 (see figure 5.5.a), we quantify the distance between these two segments as the Hausdorff distance between their skeleton curves. The Hausdorff distance is used to assign a scalar score to quantify the similarity between two IBSs. Therefore, the cost function $\gamma(s)$ for substitution operation comes down to compute the maximum distance between the points defining skeleton curve of nodes of IBSs.

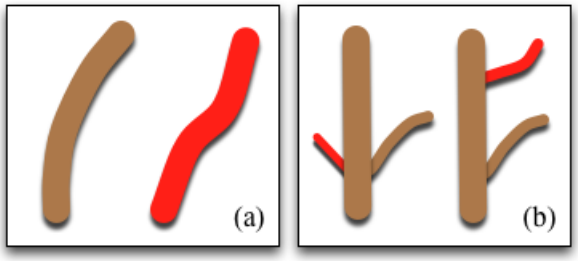


Figure 5.5: Comparison of reference and reconstructed branching patterns. a) The comparison between two IBSs is made using Hausdorff distance. Mapped segments are in red. b) Deletion and Insertion of an IBS..

Let's C_i and C_j be two skeletal polylines defined by the set of control points $\{P_k^i, k \in [0, N^i]\}$ and $\{P_l^j, l \in [0, N^j]\}$ respectively and parameterized with an index $u \in [0, 1]$. The approximation of distance between points P_i and a curve C_j is computed as follows:

$$d(P_i, C_j) = \min_u(\|P_i - C_j(u)\|) \tag{5.7}$$

The Hausdorff distance of the segments between C_i and C_j is defined as

$$d_H(C_i, C_j) = \max(\max_{k\in[0,N^i]}(d(P_i^1, C_j)), \max_{l\in[0,N^j]}(d(C_i, P_l^j))) \tag{5.8}$$

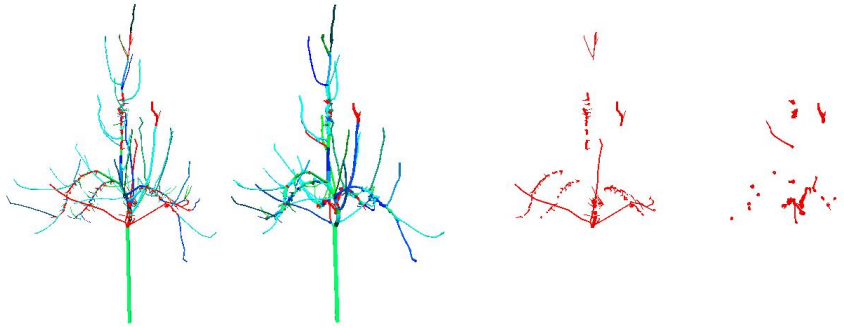
During the mapping between the two structures, some element in the reference skeleton may have no counterpart in the reference and reconstructed structure (see figure 5.5.b), we say that we have respectively a deletion or an insertion with respect the reference structure and a distance $d_H(C_i, \lambda)$ and $d_H(\lambda, C_j)$ is assigned to these special mappings respectively. The distance is defined as the size of the skeleton curve.

5.4.3 First results

Our first evaluation procedure is based on the topological comparison method (section 5.3), that we parameterize with geometrical information (section 5.4.2). This is done by calculating the minimum cost of edit operations to transform first tree into second one with cost functions weighted by geometrical comparison between individual segment.

This evaluation procedure is applied to a digitized apple tree virtually scanned afterward. The result are illustrated in figure 5.6. To visualize the resulting matching, we used red color to represent unmatched components and other colors to show matched components. Distance between the reference and reconstructed structures show that 90% of the geometry of the tree has been correctly reconstructed. Generally, most of the non-matched components are small segments, plus few big branches.

We found that two main constraints of the algorithm has strong undesired effect on the comparison. First, the distance between the plants is influenced by the



(a) (b)

Figure 5.6: The snapshots of geometrical and topological evaluation in (a) and only non-matched components are separated and showed in (b). The reference model is on left side both (a) and (b). The reconstruction model, on the other hand, is on right side

difference of their number of vertices. Therefore, the distance between T_1 and T_2 is large when the difference in the number of vertices of the given plants. Second, the topological structures between T_1 and T_2 overestimate the difference because of simple topological mistake. Only mappings that preserves the *ancestor relationship* between elements of the structure are considered by the method. As figure 5.7, branching system of T_1 and T_2 have similar topological structures but lateral branches differ in the connection of their fathers. In this case, topology of the tree will not be comparable with this method.

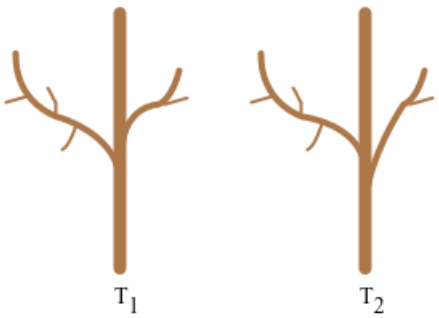


Figure 5.7: The evaluation overestimate the difference of identical topological structures.

5.5 Plant comparison based on geometrical criteria

To overcome the limitation of the previous algorithm (see Section 5.4), we propose a new algorithm taking into account constraints derived from the mapping between

the two structures. The general idea of this new approach is to first determined the mapping between elements using spatial comparison. Element that lie on similar location in space have big chance to be similar on the structures. Thus, a spatial comparison of elements gives us some correspondance between the 2 structures and no topology is required. Once mapping between elements determined, a topological comparison can be assessed by evaluating if position in the structure of each mapped elements is similar. For this, a comparison between edges of the 2 graphs is carried out.

The main problem with this approach is that the association between element using a spatial comparigon may be often ambiguous. For each segment of the reconstructed structure, one can associate in fact a list of candidate segments in the reference structure.

We formalize the problem of mapping IBSs as an optimal flow problem. Let us call T_1 and T_2 the reference and reconstructed skeletons respectively. For this, we define a bipartite graph made of two set of nodes $\{n_i \in T_1\}$ and $\{n_j \in T_2\}$ and such that any node n_i is connected to its potential candidates in T_2 and reciprocally, any node n_j is connected to its potential candidates in T_1. To decide if a connection between any two nodes n_i, n_j should exist, the minimal distance between the bounding boxes of the two elements is checked to be inferior to a threshold representing the maximal distance between elements.

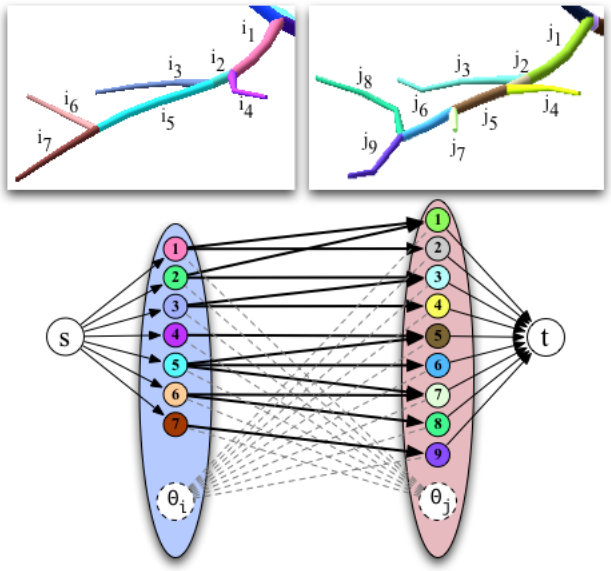


Figure 5.8: Formalisation of the problem of mapping IBS n_i of the reference tree to IBS n_j of the reconstruted tree as an optimal flow problem.

These edges are then attached a cost that corresponds to the distance between the curves corresponding to the connected nodes, (see figure 5.8). Note that two nodes θ_i and θ_j have been introduced to express the possibilities of deletion and

insertion of nodes, with their corresponding costs as explain above. Two extra nodes s and t are added and represent source and sink for the flow. Finally, we define capacities of value 1 on edges connecting s with n_i and n_j with t which define the amount of flow that should pass on these edges. This ensures that the flow will be exactly 1 on these edges and thus define a valid mapping between n_i and n_j.

The deletion and insertion costs are chosen equal to the size of the global bounding box of the model. This guarantees that a mapping between two elements is prefered over their deletion or insertion. This problem is solved using Tarjan [Tarjan 1983] extension of Edmonds and Karp's algorithm.

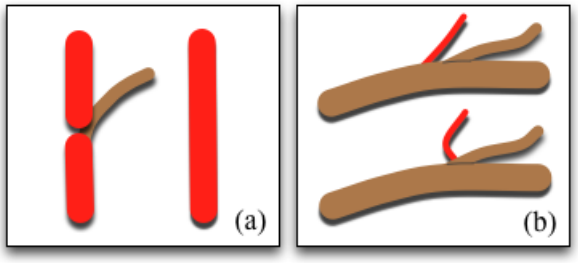


Figure 5.9: Comparison of reference and reconstructed branching patterns. a) Mapping of two IBSs separated by a branching point onto one IBS. b) Topological error in the reconstruction: the parent segment of a given IBS in one structure does not correspond to the parent segment of the corresponding IBS in the mapping (in red).

In the resulting optimal mapping M^*, some elements may be deleted or inserted. (see figure 5.9.a). However, some of these insertions/deletions may simply result from the fact that several segments in one tree altogether cover a single segment in the other tree. In this case, M^* only contains a mapping from one of the small segments to the single segment and considers that all the other small segments have been deleted (or inserted). To take this into account, a post processing step refines the mapping produced from the previous step. Each mapping of a n_i with a n_j is examined to check if one of the two elements has some neigboring elements not assigned. If for instance n_i has such a n_k neighbor, the union of the skeleton of n_i and n_k is tested with the hausdorff distance to the skeleton of n_j. If this distance value is inferior to the distance between n_i and n_j, the mapping is corrected and (i, j) is replaced by $((i, k), j)$, meaning that j is mapped on both i and k. This procedure is repeated recursively to test any sequence of n_i or n_j, thus leading to a modified, more precise mapping between T_1 and T_2.

Finally, to evaluate the difference of topology between the structures, we then inspect whether IBSs are connected in a similar way in their respective skeletal structures (see figure 5.9.b). For this, we check if an edge between two nodes n_i of the tree graph T_1 has its counterpart in T_2 that links the corresponding n_j elements. The number of deleted and inserted edges is counted and gives, in addition to the number of deleted and inserted nodes, an estimator for the accuracy of the

topological structure.

5.6 Results of the evaluation

Our evaluation procedure was applied to the different reconstructed trees presented in this thesis and results are given in Table 5.1. From the previous analysis, two indices that measure their accuracy have been defined and are presented here. From the geometrical comparison, we can determine how many branch units have been correctly identified by the reconstruction i.e. how many have a mapping onto the reference structure. The second column of the table gives the percentage of identified elements as the number of elements weighted by their sizes. The third column gives the percentage of mapped edges between the two tree graphs T_i and T_j.

	Geometry	Topology
Tree 1 (Fig 4.26 topleft)	0.98	0.95
Tree 2 (Fig 4.25 bottom)	0.96	0.62
Tree 3 (Fig 4.25 top)	0.93	0.67
Tree 4 (Fig 4.26 bottom)	0.92	0.64
Tree 5 (Fig 4.26 topright)	0.93	0.63
Cherry Tree (Fig 4.27)	0.93	0.65

Table 5.1: Evaluation of the reconstructed models.

As shown by these results, our reconstruction achieves to identify most of the tree features identified by an expert in the 3D scan. Most errors are due to small sets of outliers that make extra small branches grow in our procedure. However, some topological errors may also occur. Most of them are due to small shifts and inversions at the beginning of lateral branches, see figure 5.10. Less frequently, some branches may cross very closely. In this case, our point tracking procedure will favor one branch over the other, and will follow the path of points of the two branches.

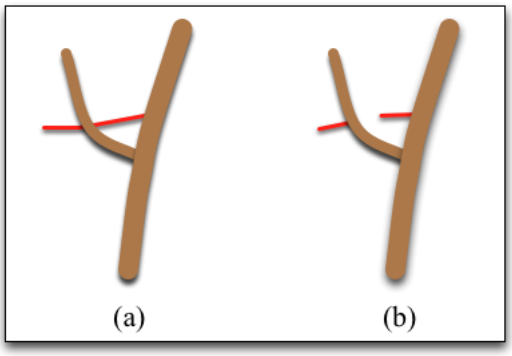


Figure 5.10: An error of topological comparison: (a) reference tree with three branches, (b) reconstructed tree with a small lateral branch.

Conclusion

In this chapter, we presented an evaluation methodology for comparing plants according to both topology and geometry. Such a method gives a strong meaning to the concept of similarity between plants. The results of these evaluations confirm the appropriateness of the proposed reconstruction pipeline. This evaluation also identified the current limitations of our reconstruction method. In particular, wrong topology may be produced in case of crossing of branches. To solve this, a global optimization of the structure could be done as a post process to find possible errors of connections between elements.

Summary

The aim and motivation of the research presented in this thesis, as outlined in the Introduction, was to develop a plant reconstruction method for 3D points cloud predominantly captured by laser scanners. 3D laser range scanning has proven to be a fast and effective way to capture the surface of an object in a computer. We used it to acquire real world trees.

However, the main bottleneck in the acquisition when scannig the surface of large trees is the variety of precision in the resulting scan ranging from dense to sparse or occluded regions. We designed a reconstruction pipeline from point set, despite of the locally sparse data, to 3D model. Based on a Riemannian graph generated from the point set, we defined a local neighborhood of points, making it possible to determine local density and orientation of the pointset. An adaptive contraction procedure enables us to concentrate points near the center of the branches. The point tracking procedure is then used to extract the skeleton of the tree. The last step is partially inspired from a method for simulating tree growth, but we modify it to achieve the generation of actual plant.

A quantitative evaluation of the reconstructed architectures against expert models trees is also described that determine similarities and difference between the two models. We found that evaluation of our models are globally goods even on complex trees. This evaluation also identified the current limitations of our reconstruction method. In particular, wrong topology may be produced in case of crossing of branches. To solve this, a global optimization of the structure could be done as a post process to find possible errors of connections between elements. Our method is sensitive to the values of its parameters. However, comparison with expert data makes it possible also to optimize these values that can thus be reused on similar data sets.

Throughout this thesis, we presented several contributions to 3D plant modeling that are finally made available to the community. The reconstruction and the evaluation methods developed during this thesis are indeed integrated to the OpenAlea platform as open-software modules and will be part of the next OpenAlea public release. They can thus be used for further studies based on the architecture of plants.

Interesting directions for future investigation are: (i) an automatic algorithm for removing noise and resolving gaps of scanned data. (ii) an automatic adjustment of the parameters for generating branching structures. (iii) a better optimization algorithm in the matching process of the evaluation procedure that would be able to mach optimally several components of one tree onto one of the other leading to even more precise evaluation.

Finally, using such tools for reconstructing a wide range of plant should make it possible to study precisely plant architecture plasticity according to environmental changes or genetic variations.

Bibliography

[Abdelhafiz 2009] A. Abdelhafiz. Integrating digital photogrammetry and terrestrial laser scanning. Geodätische Schriftenreihe der Technischen Universität Braunschweig. Inst. für Geodäsie und Photogrammetrie, 2009. (Cited on pages 13, 14 and 15.)

[Adalsteinsson & Sethian 1995] David Adalsteinsson and James A. Sethian. *A fast level set method for propagating interfaces.* J. Comput. Phys., vol. 118, pages 269–277, May 1995. (Cited on page 25.)

[Adam *et al.* 1999] B. Adam, H. Sinoquet, C. Godin and N. Donès. *3A-Software for the acquisition of plant architecture.* INRA, Clermotn-Ferrand, 1999. (Cited on page 57.)

[Adams 2006] Bart Adams. *Point-based modeling, animation and rendering of dynamic objects.* PhD thesis, Katholieke Universiteit Leuven, 2006. (Cited on page 3.)

[Alexa & Adamson 2004] Marc Alexa and Anders Adamson. *On Normals and Projection Operators for Surfaces Defined by Point Sets.* In SPBG'04 Symposium on Point-Based Graphics, pages 149–155. Eurographics Association, 2004. (Cited on page 22.)

[Alexa *et al.* 2001] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin and Claudio T. Silva. *Point set surfaces.* In Proceedings of the conference on Visualization '01, VIS '01, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society. (Cited on pages 21, 22 and 23.)

[Alexa *et al.* 2003] Marc Alexa, Johannes Behr, Daniel Cohen-or, Shachar Fleishman, David Levin and Claudio T. Silva. *Computing and rendering point set surfaces.* IEEE Transactions on Visualization and Computer Graphics, vol. 9, pages 3–15, 2003. (Cited on page 21.)

[Andersson *et al.* 2004] M. Andersson, J. Giesen, M. Pauly and B. Speckmann. *Bounds on k-Neighborhood for Locally Uniformly Sampled Surfaces.* In Proceedings of Symposium on Point-Based Graphics 04, pages 167–171, June 2004. (Cited on page 17.)

[Arya & Mount 1993] Sunil Arya and David M. Mount. *Algorithms for Fast Vector Quantization.* In Proc. of DCC '93: Data Compression Conference, pages 381–390. IEEE Press, 1993. (Cited on page 20.)

[Au *et al.* 2008] Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or and Tong-Yee Lee. *Skeleton extraction by mesh contraction.* ACM Trans. Graph., vol. 27, pages 44:1–44:10, August 2008. (Cited on page 29.)

[Aujay *et al.* 2007] Grégoire Aujay, Franck Hétroy, Francis Lazarus and Christine Depraz. *Harmonic skeleton for realistic character animation.* In Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '07, pages 151–160, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association. (Cited on page 27.)

[Baker 1995] C.J. Baker. *The development of a theoretical model for the windthrow of plants.* Theor. Biol., vol. 175, no. 3, pages 355–372, 1995. (Cited on page 32.)

[Balis *et al.* 2004] Vaios Balis, Spyros Karamitsos, Ioannis Kotsis and Christos Liapakis. *3D - Laser Scanning: Integration of Point Cloud and CCD Camera Video Data for the Production of High Resolution and Precision RGB Textured Models: Archaeological Monuments Surveying Application in Ancient Ilida.* xxx, 2004. (Cited on page 12.)

[Bentley 1975] Jon Louis Bentley. *Multidimensional binary search trees used for associative searching.* Communication of the ACM, vol. 18, pages 509–517, September 1975. (Cited on page 20.)

[Besl & McKay 1992] Paul J. Besl and Neil D. McKay. *A Method for Registration of 3-D Shapes.* IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, pages 239–256, February 1992. (Cited on page 16.)

[Biernacki *et al.* 2001] C. Biernacki, G. Celeux and G. Govaert. *Assessing a Mixture Model for Clustering with the Integrated Completed Likelihood.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 7, pages 719–725, 2001. (Cited on page 88.)

[Biernacki *et al.* 2006] Christophe Biernacki, Gilles Celeux, Gérard Govaert and Florent Langrognet. *Model-based cluster and discriminant analysis with the MIXMOD software.* Comput. Stat. Data Anal., vol. 51, no. 2, pages 587–600, November 2006. (Cited on page 88.)

[Birnbaum 1997] P. Birnbaum. *Modalités d'occupation de l'espace par les arbres en forêts guyanaise*, 1997. (Cited on pages 33 and 34.)

[Bloomenthal & Shoemake 1991] J. Bloomenthal and K. Shoemake. *Convolution surfaces.* SIGGRAPH Comput. Graph., vol. 25, pages 251–256, July 1991. (Cited on page 48.)

[Bloomenthal & Wyvill 1990] Jules Bloomenthal and Brian Wyvill. *Interactive techniques for implicit modeling.* SIGGRAPH Comput. Graph., vol. 24, pages 109–116, February 1990. (Cited on page 48.)

[Bloomenthal 1985] Jules Bloomenthal. *Modeling the mighty maple.* SIGGRAPH Comput. Graph., vol. 19, no. 3, pages 305–311, 1985. (Cited on pages 46, 47 and 90.)

[Bloomenthal 1995] J. Bloomenthal. *Skeletal design of natural forms.* PhD thesis, University of Calgary, Calgary, Alta., Canada, Canada, 1995. (Cited on page 46.)

[Borgefors 1986] Gunilla Borgefors. *Distance transformations in digital images.* Comput. Vision Graph. Image Process., vol. 34, pages 344–371, June 1986. (Cited on page 25.)

[Boudon *et al.* 2003] F. Boudon, P. Prusinkiewicz, P. Federl, C. Godin and R. Karwowski. *Interactive design of bonsai tree models.* Computer Graphics Forum. Proceedings of Eurographics, vol. 22, no. 3, pages 591–599, 2003. (Cited on pages 49 and 51.)

[Boudon *et al.* 2006] F. Boudon, A. Meyer and C. Godin. *Survey on Computer Representations of Trees for Realistic and Efficient Rendering.* Rapport de recherche 2301, LIRIS, Université Claude Bernard Lyon 1, 2006. (Cited on pages 45 and 51.)

[Boudon *et al.* 2012] F. Boudon, C. Pradal, T. Cokelaer, P. Prusinkiewicz and C. Godin. *L-Py: an L-System simulation framework for modeling plant development based on a dynamic language.* Frontiers in Plant Science, vol. 3, no. 76, 2012. (Cited on page 91.)

[Boudon 2004] F. Boudon. *Représentation géométrique de l'architecture des plantes.* PhD thesis, Université de Montpellier II, feb 2004. (Cited on pages 45, 50 and 75.)

[Cao *et al.* 2010] Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang and Zhinxun Su. *Point Cloud Skeletons via Laplacian Based Contraction.* In Proceedings of the 2010 Shape Modeling International Conference, SMI '10, pages 187–197, Washington, DC, USA, 2010. IEEE Computer Society. (Cited on pages 29 and 82.)

[Casula *et al.* 2007] G. Casula, S. Fais, P. Ligas and P. Mora. *Experimental application of a 3D terrestrial laser scanner and acoustic techniques in assessing the quality of the stones used in monumental structures.* In 4th International Conference for NDT, 2007. (Cited on page 72.)

[Cescatti 1997] A. Cescatti. *Modelling the radiative transfer in discontinuous canopies of asymmetric crowns. I. Model structure and algorithms.* Ecological Modelling, vol. 101, no. 2, pages 263–274, August 1997. (Cited on page 32.)

[Cornea *et al.* 2007] N.D. Cornea, D. Silver and P. Min. *Curve-Skeleton Properties, Applications, and Algorithms.* Visualization and Computer Graphics, IEEE Transactions on, vol. 13, no. 3, pages 530 –548, may-june 2007. (Cited on pages 23, 24, 25 and 26.)

[Costes *et al.* 2003] E. Costes, H. Sinoquet, J.J. Kelner and C. Godin. *Exploring within-tree architectural development of two apple tree cultivars over 6 years.* Annals of Botany, vol. 91, pages 91–104, 2003. (Cited on pages 38, 57 and 72.)

[Côté *et al.* 2009] Jean-François Côté, Jean-Luc Widlowski, Richard A. Fournier and Michel M. Verstraete. *The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial lidar.* Remote Sensing of Environment, vol. 113, no. 5, pages 1067 – 1081, 2009. (Cited on pages 66, 95 and 96.)

[Da Silva 2008] D. Da Silva. *Caractérisation de la nature multi-échelles des plantes par des outils de l'analyse fractale, application à la modélisation de l'interception de la lumière.* PhD thesis, Université Montpellier 2, 2008. (Cited on pages 38 and 39.)

[Dassot *et al.* 2011] Mathieu Dassot, Thiéry Constant and Meriem Fournier. *The use of terrestrial LiDAR technology in forest science: application fields, benefits and challenges.* Annals of Forest Science, pages 1–16, 2011. (Cited on page 11.)

[David Barber & Bryan 2001] Dr Jon Mills David Barber and Paul Bryan. *Laser scanning and photogrammetry: 21st century metrology.* In Proceedings of 18th International Symposium CIPA 2001, pages 360–366, September 2001. (Cited on page 13.)

[de Reffye *et al.* 1988] Phillippe de Reffye, Claude Edelin, Jean Françon, Marc Jaeger and Claude Puech. *Plant models faithful to botanical structure and development.* SIGGRAPH Comput. Graph., vol. 22, pages 151–158, June 1988. (Cited on pages 39 and 45.)

[Deleuze & Houllier 1997] C. Deleuze and F. Houllier. *A transport model for tree ring width.* Silva Fenn, vol. 31, pages 239–250, 1997. (Cited on page 36.)

[Deussen & Lintermann 2005] O. Deussen and B. Lintermann. Digital design of nature. Springer-Verlag, 2005. (Cited on page 71.)

[Deussen *et al.* 2002] O. Deussen, C. Colditz, M. Stamminger and G. Drettakis. *Interactive visualization of complex plant ecosystems.* In Proceedings of the conference on Visualization '02, VIS '02, pages 219–226, Washington, DC, USA, 2002. IEEE Computer Society. (Cited on pages 52 and 53.)

[Dijkstra 1959] Edsger W. Dijkstra. *A note on two problems in connexion with graphs.* Numerische Mathematik, vol. 1, pages 269–271, 1959. (Cited on page 78.)

[Edelsbrunner & Mücke 1994] Herbert Edelsbrunner and Ernst P. Mücke. *Three-dimensional alpha shapes.* ACM Trans. Graph., vol. 13, no. 1, pages 43–72, 1994. (Cited on page 68.)

[Elkhrachy 2008] Ismail Elkhrachy. *Towards an automatic registration for terrestrial laser scanner data.* PhD thesis, Technischen Universitá Carolo-Wilhelmina zu Braunschweig, 2008. (Cited on pages 12 and 14.)

[Evers *et al.* 2005] J.B. Evers, J. Vos, C. Fournier, B. Andrieu, M. Chelle and P.C. Struik. *Towards a generic architectural model of tillering in Gramineae, as exemplified by spring wheat (Triticum aestivum).* New Phytologist, vol. 166, no. 3, pages 801–812, 2005. (Cited on pages 56 and 57.)

[Farque *et al.* 2001] L Farque, H Sinoquet and F Colin. *Canopy structure and light interception in Quercus petraea seedlings in relation to light regime and plant density.* Tree Physiology, vol. 21, no. 17, pages 1257–67, 2001. (Cited on page 57.)

[Ferraro & Godin 2000] P. Ferraro and C. Godin. *A distance measure between plant architectures.* Annals of Forest Science, vol. 57, no. 5/6, pages 445–461, 2000. (Cited on pages 95, 97, 98 and 100.)

[Ferraro 2000] P. Ferraro. *Méthodes algorithmiques de comparaison d'arborescences. Applications à la comparaison de l'architecture des plantes.* PhD thesis, Institut National Polytechnique de Toulouse, November 2000. (Cited on page 43.)

[Friedman *et al.* 1977] Jerome H. Friedman, Jon Louis Bentley and Raphael Ari Finkel. *An Algorithm for Finding Best Matches in Logarithmic Expected Time.* ACM Transactions on Mathematical Software, vol. 3, pages 209–226, September 1977. (Cited on page 20.)

[Galbraith *et al.* 2004] Callum Galbraith, Peter MacMurchy and Brian Wyvill. *BlobTree Trees.* In Proceedings of the Computer Graphics International, CGI '04, pages 78–85, Washington, DC, USA, 2004. IEEE Computer Society. (Cited on page 48.)

[Giannitrapani & Murino 1999] Riccardo Giannitrapani and Vittorio Murino. *Three-Dimensional Skeleton Extraction by Point Set Contraction.* In ICIP (1), pages 565–569, 1999. (Cited on pages 9 and 82.)

[Godin & Caraglio 1998] C. Godin and Y. Caraglio. *A multiscale model of plant topological structures.* Journal of Theoretical Biology, vol. 191, pages 1–46, 1998. (Cited on pages 40, 41, 42, 44 and 49.)

[Godin *et al.* 1999] C. Godin, E. Costes and H. Sinoquet. *A Method for Describing Plant Architecture which Integrates Topology and Geometry.* Annals of Botany, vol. 84, no. 3, pages 343–357, 1999. (Cited on page 57.)

[Godin *et al.* 2005] C. Godin, E. Costes and H. Sinoquet. *Plant architecture modelling - virtual plants and complex systems.* In C. Turnbull, editeur, Plant Architecture and its Manipulation, volume 17 of *Annual plant reviews*, pages 238–287. Blackwell, 2005. (Cited on pages 43 and 56.)

[Godin 2000] C. Godin. *Representing and encoding plant architecture: A review.* Annals of Forest Science, vol. 57, no. 5, pages 413–438, June 2000. (Cited on pages 31, 36, 37, 39 and 41.)

[Gorte & Pfeifer 2004] Ben Gorte and Norbert Pfeifer. *Structuring laser-scanned trees using 3D mathematical morphology.* International Archives of Photogrammetry and Remote Sensing, vol. 35, no. B5, page 929ï¿½933, 2004. (Cited on page 65.)

[Goshtasby 1989] A. Goshtasby. *Stereo Correspondence by Selective Search.* In Proc. Japan Computer Vision Conf, pages 1–10, July 1989. (Cited on page 7.)

[Greene 1989] N. Greene. *Voxel space automata: modeling with stochastic growth processes in voxel space.* SIGGRAPH Comput. Graph., vol. 23, pages 175–184, July 1989. (Cited on page 38.)

[Guennebaud & Gross 2007] Gaël Guennebaud and Markus Gross. *Algebraic point set surfaces.* ACM Trans. Graph., vol. 26, July 2007. (Cited on page 22.)

[Hallé *et al.* 1978] F. Hallé, R. A. A. Oldeman and P. B. Tomlinson. Tropical trees and forests - an architectural analysis. Springer Verlag, 1978. (Cited on page 31.)

[Hanan & Wang 2004] J.S. Hanan and Y. Wang. *Floradig: a configurable program for capturing plant architecture.* In Proceedings of the 4th International Workshop on Functional-Structural Plant Models, pages 407–411, 2004. (Cited on page 57.)

[Hasler *et al.* 2010] Nils Hasler, Thorsten Thormählen, Bodo Rosenhahn and Hans-Peter Seidel. *Learning skeletons for shape and pose.* In Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games, I3D '10, pages 23–30, New York, NY, USA, 2010. ACM. (Cited on page 27.)

[He *et al.* 2009] Ying He, Xian Xiao and Hock-Soon Seah. *Harmonic 1-form based skeleton extraction from examples.* Graph. Models, vol. 71, pages 49–62, March 2009. (Cited on page 27.)

[Honda 1971] Hisao Honda. *Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body.* Journal of Theoretical Biology, vol. 31, no. 2, pages 331–338, May 1971. (Cited on page 39.)

[Hoppe *et al.* 1992] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald and Werner Stuetzle. *Surface reconstruction from unorganized points.* SIGGRAPH Comput. Graph., vol. 26, pages 71–78, July 1992. (Cited on pages 18 and 81.)

[Horn 1971] H. Horn. The adaptive geometry of trees. Princeton University Press, Princeton, N.J., 1971. (Cited on page 32.)

[Jarvis 1983] R.A. Jarvis. *A Perspective on Range Finding Techniques for Computer Vision.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-5, pages 122–139, 1983. (Cited on page 5.)

[Johnson & Wichern 2002] R. A. Johnson and D. W. Wichern, editeurs. Applied multivariate statistical analysis. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 5th édition, 2002. (Cited on page 81.)

[Katz & Tal 2003] Sagi Katz and Ayellet Tal. *Hierarchical mesh decomposition using fuzzy clustering and cuts.* ACM Trans. Graph., vol. 22, pages 954–961, July 2003. (Cited on page 26.)

[Kawaguchi 1982] Yoichiro Kawaguchi. *A morphological study of the form of nature.* SIGGRAPH Comput. Graph., vol. 16, pages 223–232, July 1982. (Cited on pages 45 and 46.)

[Kobbelt & Botsch 2004] Leif Kobbelt and Mario Botsch. *A survey of point-based techniques in computer graphics.* Comput. Graph., vol. 28, pages 801–814, December 2004. (Cited on page 3.)

[Kong & Rosenfeld 1989] T. Y. Kong and A. Rosenfeld. *Digital topology: introduction and survey.* Comput. Vision Graph. Image Process., vol. 48, pages 357–393, December 1989. (Cited on page 23.)

[Koop 1989] H. Koop. Silvi-star: A comprehensive monitoring system. Forest Dynamics, 1989. (Cited on page 32.)

[Lang 1973] A. R. G. Lang. *Leaf orientation of a cotton crop.* Agricultural and Forest Meteorology, vol. 11, pages 37–51, 1973. (Cited on page 55.)

[Lee *et al.* 2001] K.H. Lee, H. Park and S. Son. *A Framework for Laser Scan Planning of Freeform Surfaces.* The International Journal of Advanced Manufacturing Technology, vol. 17, pages 171–180, 2001. 10.1007/s001700170187. (Cited on page 4.)

[Levin 1998] David Levin. *The Approximation Power Of Moving Least-Squares.* Mathematics of Computation, vol. 67, pages 1517–1531, 1998. (Cited on page 21.)

[Levin 2003] David Levin. *Mesh-independent surface interpolation.* Geometric Modeling for Scientific Visualization, pages 37–49, 2003. (Cited on page 21.)

[Lhuillier & Quan 2005] M. Lhuillier and L. Quan. *A quasi-dense approach to surface reconstruction from uncalibrated images.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 27, no. 3, pages 418 –433, march 2005. (Cited on page 62.)

[Li *et al.* 2010] Guo Li, Ligang Liu, Hanlin Zheng and Niloy J. Mitra. *Analysis, reconstruction and manipulation using arterial snakes.* ACM Trans. Graph., vol. 29, pages 152:1–152:10, December 2010. (Cited on pages 28 and 29.)

[Lien *et al.* 2006] Jyh-Ming Lien, John Keyser and Nancy M. Amato. *Simultaneous shape decomposition and skeletonization.* In Proceedings of the 2006 ACM symposium on Solid and physical modeling, SPM '06, pages 219–228, New York, NY, USA, 2006. ACM. (Cited on page 26.)

[Lieutier 2003] André Lieutier. *Any open bounded subset of Rn has the same homotopy type than its medial axis.* In Proceedings of the eighth ACM symposium on Solid modeling and applications, SM '03, pages 65–75, New York, NY, USA, 2003. ACM. (Cited on page 23.)

[Lin *et al.* 2005] Yan-Ping Lin, Cheng-Tao Wang and Ke-Rong Dai. *Reverse engineering in CAD model reconstruction of customized artificial joint.* Medical Engineering Physics, vol. 27, no. 2, pages 189 – 193, 2005. (Cited on page 4.)

[Linsen 2001] Lars Linsen. *Point Cloud Representation.* Rapport technique, Faculty of Computer Science, University of Karlsruhe, 2001. (Cited on page 18.)

[Liu *et al.* 2003] Pin-Chou Liu, Fu-Che Wu, Wan-Chun Ma, Rung-Huei Liang and Ming Ouhyoung. *Automatic Animation Skeleton Construction Using Repulsive Force Field.* In Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, PG '03, pages 409–, Washington, DC, USA, 2003. IEEE Computer Society. (Cited on page 26.)

[Livny *et al.* 2010] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang and Jihad El-sana. *Automatic Reconstruction of Tree Skeletal Structures from Point Clouds.* ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2010), vol. 29, pages 151:1–151:8, 2010. (Cited on page 67.)

[Livny *et al.* 2011] Yotam Livny, Soeren Pirk, Zhanglin Cheng, Feilong Yan, Oliver Deussen, Daniel Cohen-Or and Baoquan Chen. *Texture-lobes for tree modelling.* ACM Trans. Graph., vol. 30, pages 53:1–53:10, August 2011. (Cited on pages 67 and 68.)

[Lloyd 1982] Stuart P. Lloyd. *Least Squares Quantization in PCM.* IEEE Transactions on Information Theory, vol. IT-28, no. 2, pages 129–137, March 1982. (Cited on pages 22 and 66.)

[Lu 1979] Shin Y. Lu. *A tree-to-tree distance and its application to cluster Analysis.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 1, pages 219–224, 1979. (Cited on page 97.)

[Ma & Wan 2001] Cherng-Min Ma and Shu-Yen Wan. *A medial-surface oriented 3-d two-subfield thinning algorithm.* Pattern Recognition Letters, vol. 22, no. 13, pages 1439 – 1446, 2001. (Cited on page 25.)

[Macgovern & Wyant 1971] A J Macgovern and J C Wyant. *Computer generated holograms for testing optical elements.* Applied Optics, vol. 10, no. 3, pages 619–24, 1971. (Cited on page 6.)

[Moulia & Sinoquet 1993] B. Moulia and H. Sinoquet. *Three-dimensional digitizing systems for plants canopy geometrical structure: a review.* In C. Varlet-Grancher, R. Bonhomme and H. Sinoquet, editeurs, Crop structure and light microclimate: Characterization and applications, Science Update. INRA, 147 rue de l'Université, 75338 Paris cedex 07, France, 1993. (Cited on page 55.)

[Neubert *et al.* 2007] Boris Neubert, Thomas Franken and Oliver Deussen. *Approximate image-based tree-modeling using particle flows.* ACM Trans. Graph., vol. 26, July 2007. (Cited on pages 39, 62, 63 and 64.)

[Norman & Welles 1983] J M Norman and J M Welles. *Radiative transfer in an array of canopies.* Agronomy Journal, vol. 75, no. 3, pages 481–488, 1983. (Cited on page 32.)

[Okabe *et al.* 2000] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara and Sung Nok Chiu. Spatial tessellations: Concepts and applications of Voronoi diagrams. Probability and Statistics. Wiley, NYC, 2nd édition, 2000. 671 pages. (Cited on page 22.)

[Okabe *et al.* 2006] M. Okabe, S. Owada and T. Igarashi. *Interactive design of botanical trees using freehand sketches and example-based editing.* In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, New York, NY, USA, 2006. ACM. (Cited on pages 58 and 59.)

[Palágyi *et al.* 2001] Kálmán Palágyi, Erich Sorantin, Emese Balogh, Attila Kuba, Csongor Halmai, Balázs Erdohelyi and Klaus Hausegger. *A Sequential 3D Thinning Algorithm and Its Medical Applications.* In Proceedings of the 17th International Conference on Information Processing in Medical Imaging, IPMI '01, pages 409–415, London, UK, 2001. Springer-Verlag. (Cited on page 65.)

[Palubicki *et al.* 2009] Wojciech Palubicki, Kipp Horel, Steven Longay, Adam Runions, Brendan Lane, Radomír Měch and Przemyslaw Prusinkiewicz. *Self-organizing tree models for image synthesis.* In SIGGRAPH '09: ACM SIGGRAPH 2009 papers, pages 1–10, New York, NY, USA, 2009. ACM. (Cited on pages 60 and 84.)

[Park & Chang 2009] Sang C. Park and Minho Chang. *Reverse engineering with a structured light system.* Computers Industrial Engineering, vol. 57, no. 4, pages 1377 – 1384, 2009. (Cited on pages 5 and 7.)

[Parveaud 2006] C.-E. Parveaud. *Propriétés des couronnes de noyers (Juglans nigra x J.regia) et croissance des pousses annuelles - influence de la géométrie du*

feuillage, de la position des pousses et de leur climat radiatif. PhD thesis, Université Montpellier 2, 2006. (Cited on page 97.)

[Pauly *et al.* 2002] Mark Pauly, Markus Gross and Leif P. Kobbelt. *Efficient simplification of point-sampled surfaces.* In VIS '02: Proceedings of the conference on Visualization '02, pages 163–170. IEEE Computer Society, 2002. (Cited on pages 79 and 81.)

[Pauly 2003] Mark Pauly. *Point primitives for interactive modeling and processing of 3D geometry.* PhD thesis, ETH Zurich, 2003. (Cited on pages 17, 19 and 21.)

[Pharr & Humphreys 2004] Matt Pharr and Greg Humphreys. Physically based rendering: From theory to implementation. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. (Cited on page 96.)

[Phattaralerphong & Sinoquet 2005] J. Phattaralerphong and H. Sinoquet. *A method for 3D reconstruction of tree crown volume from photographs: assessment with 3D-digitized plants.* Tree Physiology, vol. 25, no. 10, pages 1229–1242, 2005. (Cited on pages 32, 39 and 57.)

[Piegl & Tiller 1997] L. Piegl and W. Tiller. The nurbs book (2nd ed.). Springer-Verlag New York, Inc., New York, NY, USA, 1997. (Cited on pages 35, 36 and 46.)

[Piegl & Tiller 2002] Les A. Piegl and Wayne Tiller. *Surface skinning revisited.* The Visual Computer, vol. 18, pages 273–283, 2002. (Cited on page 35.)

[Polhemus 1993] A. Polhemus. *3SPACE FASTRAK User's Manual, Revision F.* Kaiser Aerospace and Electronics Company, PO Box 560, Colchester, Vermont, 05446, USA, 1993. (Cited on page 57.)

[POV-RAY] POV-RAY. *The Persistence of Vision Raytracer.* (Cited on page 97.)

[Pradal *et al.* 2009] C. Pradal, F. Boudon, C. Nouguier, J. Chopard and C. Godin. *PlantGL : a Python-based geometric library for 3D plant modelling at different scales.* Graphical Models, vol. 71, no. 1, pages 1–21, jan 2009. (Cited on pages 32, 33, 34, 35, 45, 49, 75 and 97.)

[Prasad 1997] L. Prasad. *Morphological analysis of shapes*, 1997. (Cited on page 59.)

[Pratt 1987] Vaughan Pratt. *Direct least-squares fitting of algebraic surfaces.* SIGGRAPH Comput. Graph., vol. 21, pages 145–152, August 1987. (Cited on page 22.)

[Preparata & Shamos 1985] Franco P. Preparata and Michael I. Shamos. Computational geometry: an introduction. Springer-Verlag New York, Inc., New York, NY, USA, 1985. (Cited on page 17.)

[Prusinkiewicz & Lindenmayer 1990] P. Prusinkiewicz and A. Lindenmayer. The algorithmic beauty of plants. Springer-Verlag New York, Inc., New York, NY, USA, 1990. (Cited on pages 39, 40 and 71.)

[Prusinkiewicz *et al.* 1994] Przemyslaw Prusinkiewicz, Mark James and Radomír Měch. *Synthetic topiary.* In Proceedings of the 21st annual conference on Computer graphics and interactive techniques, SIGGRAPH '94, pages 351–358, New York, NY, USA, 1994. ACM. (Cited on page 39.)

[Quan *et al.* 2006] Long Quan, Ping Tan, Gang Zeng, Lu Yuan, Jingdong Wang and Sing Bing Kang. *Image-based plant modeling.* ACM Trans. Graph., vol. 25, pages 599–604, July 2006. (Cited on pages 55, 61 and 62.)

[Rakocevic *et al.* 2000] M. Rakocevic, H. Sinoquet, A. Christophe and C. Varlet-Grancher. *Assessing the Geometric Structure of a White Clover (Trifolium repens L.) Canopy using3-D Digitising.* Annals of Botany, vol. 86, no. 3, pages 519–526, 2000. (Cited on pages 56 and 57.)

[Reche-Martinez *et al.* 2004] Alex Reche-Martinez, Ignacio Martin and George Drettakis. *Volumetric reconstruction and interactive rendering of trees from photographs.* ACM Trans. Graph., vol. 23, pages 720–727, August 2004. (Cited on pages 32, 39 and 62.)

[Reeb 1946] G. Reeb. *On the singular points of a completely integrable pfaff form or of a numerical function.* Comptes Randus Acad., vol. 222, pages 847–849, 1946. (Cited on page 27.)

[Reeves & Blau 1985] W. T. Reeves and R. Blau. *Approximate and probabilistic algorithms for shading and rendering structured particle systems.* SIGGRAPH Comput. Graph., vol. 19, pages 313–322, July 1985. (Cited on pages 51 and 52.)

[Reshetyuk 2009] Yuriy Reshetyuk. Self-calibration and direct georeferencing in terrestrial laser scanning. KTH, Sweden, 2009. (Cited on pages 9, 14 and 15.)

[Ross 1981] J. K. Ross. The radiation regime and architecture of plant stands. W. Junk, The Hague, Netherlands., 1981. (Cited on page 31.)

[Runions *et al.* 2007] Adam Runions, Brendan Lane and Przemyslaw Prusinkiewicz. *Modeling Trees with a Space Colonization Algorithm.* In D. Ebert and S. Merillou, editeurs, Eurographics Workshop on Natural Phenomena, pages 63–70, Prague, Czech Republic, 2007. Eurographics Association. (Cited on pages 60, 61 and 84.)

[Ruzon & Tomasi 2000] M.A. Ruzon and C. Tomasi. *Alpha estimation in natural images.* In Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, volume 1, pages 18–25, 2000. (Cited on page 62.)

[Sahoo & Menq 1991] K.C. Sahoo and C.H. Menq. *Localization of 3-D objects having complex sculptured surfaces using tactile sensing and surface description.* Journal of engineering for industry, vol. 113, no. 1, pages 85–92, 1991. (Cited on page 4.)

[Samet 1990] Hanan Samet. The design and analysis of spatial data structures. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990. (Cited on page 19.)

[Savio *et al.* 2007] E. Savio, L. De Chiffre and R. Schmitt. *Metrology of freeform shaped parts.* CIRP Annals - Manufacturing Technology, vol. 56, no. 2, pages 810 – 835, 2007. (Cited on pages 9 and 10.)

[Schaefer & Yuksel 2007] S. Schaefer and C. Yuksel. *Example-based skeleton extraction.* In Proceedings of the fifth Eurographics symposium on Geometry processing, pages 153–162, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association. (Cited on page 27.)

[Schwenke *et al.* 2002] Heinrich Schwenke, Ulrich Neuschaefer-Rube, Tilo Pfeifer and Horst Kunzmann. *Optical Methods for Dimensional Metrology in Production Engineering.* CIRP Annals - Manufacturing Technology, vol. 51, no. 2, pages 685 – 699, 2002. (Cited on page 5.)

[Selkow 1977] Stanley M. Selkow. *The Tree-to-Tree Editing Problem.* Inf. Process. Lett., vol. 6, no. 6, pages 184–186, 1977. (Cited on page 97.)

[Shahram & Saeed 2006] Mohammad Nejad Shahram and Olyaee Saeed. *Comparison of TOF, FMCW and Phase-Shift Laser Range-Finding Methods by Simulation and Measurement.* Journal of Technology of Education, vol. 1, no. 1, 2006. (Cited on page 12.)

[Sharf *et al.* 2006] Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt and Daniel Cohen-Or. *Competing fronts for coarse-to-fine surface reconstruction.* In Eurographics 2006 (Computer Graphics Forum), volume 25, pages 389–398, Vienna, october 2006. Eurographics. (Cited on page 28.)

[Sharf *et al.* 2007] Andrei Sharf, Thomas Lewiner, Ariel Shamir and Leif Kobbelt. *On-the-fly curve-skeleton computation for 3d shapes.* In Eurographics 2007 (Computer Graphics Forum), volume 26, pages 323–328, Prague, october 2007. Eurographics. (Cited on pages 27 and 28.)

[Shinozaki *et al.* 1964] Kichiro Shinozaki, Kyoji Yoda, Kazuo Hozumi and Tatuo Kira. *A quantitative analysis of plant form – the pipe model theory.* Japanese Journal of Ecology, vol. 14, no. 3, pages 97–105, 1964. (Cited on page 89.)

[Shlyakhter *et al.* 2001] Ilya Shlyakhter, Max Rozenoer, Julie Dorsey and Seth Teller. *Reconstructing 3D Tree Models from Instrumented Photographs.* IEEE

Comput. Graph. Appl., vol. 21, pages 53–61, May 2001. (Cited on pages 32, 61 and 62.)

[Sillion 1995] François Sillion. *Hierarchical Solution Techniques for Realistic Rendering.* In State of the Art Report - Graphicon'95, july 1995. (Cited on page 39.)

[Sinoquet & Bonhomme 1992] Hervé Sinoquet and Raymond Bonhomme. *Modeling radiative transfer in mixed and row intercropping systems.* Agricultural and Forest Meteorology, vol. 62, no. 3-4, pages 219–240, Dec 1992. (Cited on page 38.)

[Sinoquet & Rivet 1997] H. Sinoquet and P. Rivet. *Measurement and visualization of the architecture of an adult tree based on a three-dimensional digitising device.* Trees - Structure and Function, vol. 11, no. 5, pages 265–270, April 1997. (Cited on pages 56 and 57.)

[Sinoquet *et al.* 1991] H. Sinoquet, B. Moulia and R. Bonhomme. *Estimating the three-dimensional geometry of a maize crop as an input of radiation models: comparison between three-dimensional digitizing and plant profiles.* Agricultural and Forest Meteorology, vol. 55, no. 3-4, pages 233 – 249, 1991. (Cited on page 56.)

[Sinoquet *et al.* 1997a] H. Sinoquet, P. Rivet and C. Godin. *Assessment of the three-dimensional architecture of walnut trees using digitising.* Silva Fennica, vol. 31, no. 3, pages 265–273, 1997. (Cited on page 42.)

[Sinoquet *et al.* 1997b] H. Sinoquet, P. Rivet and C. Godin. *Assessment of the three-dimensional architecture of walnut trees using digitising.* Silva Fennica, vol. 31, pages 265–273, 1997. (Cited on page 57.)

[Sonohat *et al.* 2002] G. Sonohat, H. Sinoquet, C. Varlet-Grancher, M. Rakocevic, A. Jacquet, J.-C. Simon and B. Adam. *Leaf dispersion and light partitioning in three-dimensionally digitized tall fescueâwhite clover mixtures.* Plant, Cell Environment, vol. 25, no. 4, pages 529–538, 2002. (Cited on page 57.)

[Sonohat *et al.* 2006] G. Sonohat, H. Sinoquet, V. Kulandaivelu, D. Combes and F. Lescourret. *Three-dimensional reconstruction of partially 3D-digitized peach tree canopies.* Tree Physiology, vol. 26, no. 3, pages 337–351, 2006. (Cited on page 58.)

[Sperry *et al.* 1998] J. S. Sperry, F. R. Adler, G. S. Campbell and J. P. Comstock. *Limitation of plant water use by rhizosphere and xylem conductance: results from a model.* Plant, Cell & Environment, vol. 21, no. 4, pages 347–359, 1998. (Cited on page 37.)

[Spyridi & Requicha 1990] A.J. Spyridi and A.A.G. Requicha. *Accessibility analysis for the automatic inspection of mechanical parts by coordinate measuring*

machines. In Proceedings of the IEEE International Conference on Robotics and Automation, pages 1284–1289, may 1990. (Cited on page 4.)

[Tagliasacchi *et al.* 2009] Andrea Tagliasacchi, Hao Zhang and Daniel Cohen-Or. *Curve skeleton extraction from incomplete point cloud*. ACM Trans. Graph., vol. 28, pages 71:1–71:9, July 2009. (Cited on pages 28, 29 and 82.)

[Tai 1979] Kuo-Chung Tai. *The Tree-to-Tree Correction Problem*. J. ACM, vol. 26, no. 3, pages 422–433, july 1979. (Cited on page 97.)

[Takenaka *et al.* 1998] A. Takenaka, Y. Inui and A. Osawa. *Measurement of three-dimensional structure of plants with a simple device and estimation of light capture of individual leaves*. Functional Ecology, vol. 12, no. 1, pages 159–165, 1998. (Cited on pages 55 and 56.)

[Tan *et al.* 2007] Ping Tan, Gang Zeng, Jingdong Wang, Sing Bing Kang and Long Quan. *Image-based tree modeling*. ACM Trans. Graph., vol. 26, July 2007. (Cited on pages 61, 62, 63, 64 and 67.)

[Tan *et al.* 2008] Ping Tan, Tian Fang, Jianxiong Xiao, Peng Zhao and Long Quan. *Single image tree modeling*. ACM Trans. Graph., vol. 27, no. 5, page 108, 2008. (Cited on page 64.)

[Tarjan 1983] Robert Endre Tarjan. Data structures and network algorithms. CBMS-NFS, 1983. (Cited on page 105.)

[Telea & Vilanova 2003] Alexandru Telea and Anna Vilanova. *A robust level-set algorithm for centerline extraction*. In Proceedings of the symposium on Data visualisation 2003, VISSYM '03, pages 185–194, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. (Cited on page 25.)

[Thornley 1969] J.H.M. Thornley. *A model to describe the partitioning of photosynthate during vegetative plant growth*. Ann. Botany, vol. 33, pages 419–430, 1969. (Cited on page 36.)

[Thornley 1972] J.H.M. Thornley. *A balanced quantitative model for root:shoot ratios in vegetative plants*. Ann. Botany, vol. 36, pages 431–441, 1972. (Cited on page 36.)

[Tyree & Ewers 1991] Melvin T. Tyree and Frank W. Ewers. *The hydraulic architecture of trees and other woody plants*. New Phytologist, vol. 119, no. 3, pages 345–360, 1991. (Cited on page 37.)

[Verroust & Lazarus 2000a] Anne Verroust and Francis Lazarus. *Extracting skeletal curves from 3D scattered data*. Visual Computer, vol. 16, pages 15–25, 2000. (Cited on page 66.)

[Verroust & Lazarus 2000b] Anne Verroust and Francis Lazarus. *Extracting skeletal curves from 3D scattered data.* Visual Computer, vol. 16, pages 15–25, 2000. (Cited on page 78.)

[Várady *et al.* 1997] Tamás Várady, Ralph R Martin and Jordan Cox. *Reverse engineering of geometric models–an introduction.* Computer-Aided Design, vol. 29, no. 4, pages 255 – 268, 1997. Reverse Engineering of Geometric Models. (Cited on pages 4 and 6.)

[Wagner & Fischer 1974] Robert A. Wagner and Michael J. Fischer. *The String-to-String Correction Problem.* J. ACM, vol. 21, no. 1, pages 168–173, January 1974. (Cited on pages 97 and 98.)

[Watanabe *et al.* 2005] T. Watanabe, J.S. Hanan, P.M. Room and T. Hasegawa. *Rice morphogenesis and plant architecture: measurement, specification and the reconstruction of structural development by 3D architectural modelling.* Annals of Botany, vol. 95, no. 7, pages 1131–1143, 2005. (Cited on page 56.)

[Weber & Penn 1995] J. Weber and J. Penn. *Creation and rendering of realistic trees.* In Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95, pages 119–128, New York, NY, USA, 1995. ACM. (Cited on pages 52 and 53.)

[Welzl 1991] Emo Welzl. *Smallest Enclosing Disks (balls and Ellipsoids).* In Results and New Trends in Computer Science, pages 359–370. Springer-Verlag, 1991. (Cited on page 89.)

[Widlowski *et al.* 2006] Jean-Luc Widlowski, Thomas Lavergne, Bernard Pinty, Michel Verstraete and Nadine Gobron. *Rayspread: A Virtual Laboratory for Rapid BRF Simulations Over 3-D Plant Canopies.* In Frank Graziani, editeur, Computational Methods in Transport, volume 48 of *Lecture Notes in Computational Science and Engineering*, pages 211–231. Springer Berlin Heidelberg, 2006. (Cited on page 96.)

[Will & Pennington 1972] P.M. Will and K.S. Pennington. *Grid coding: A novel technique for image processing.* Proceedings of the IEEE, vol. 60, no. 6, pages 669–680, june 1972. (Cited on page 7.)

[Wither *et al.* 2009] J. Wither, F. Boudon, M.-P. Cani and C. Godin. *Structure from silhouettes: a new paradigm for fast sketch-based design of trees.* In Computer Graphic Forum. Special issue: Eurographics 2009, volume 28 (2), page 541â550, 2009. (Cited on pages 59 and 60.)

[Wolf *et al.* 2000] Kai Wolf, Dieter Roller and Dirk Schäfer. *An approach to computer-aided quality control based on 3D coordinate metrology.* Journal of Materials Processing Technology, vol. 107, no. 1-3, pages 96 – 110, 2000. (Cited on pages 4 and 6.)

[Woodward 1988] C. D. Woodward. *Skinning techniques for interactive B-spline surface interpolation.* Comput. Aided Des., vol. 20, pages 441–451, October 1988. (Cited on page 35.)

[Wu *et al.* 2006] Fu-Che Wu, Wan-Chun Ma, Rung-Huei Liang, Bing-Yu Chen and Ming Ouhyoung. *Domain connected graph: the skeleton of a closed 3D shape for animation.* Vis. Comput., vol. 22, pages 117–135, February 2006. (Cited on page 27.)

[Xu *et al.* 2007] Hui Xu, Nathan Gossett and Baoquan Chen. *Knowledge and heuristic-based modeling of laser-scanned trees.* ACM Trans. Graph., vol. 26, no. 4, page 19, 2007. (Cited on pages 63, 65, 66 and 67.)

[Yan *et al.* 2009] D-M Yan, J. Wintz, B. Mourrain, W. Wang, F. Boudon and C. Godin. *Efficient and robust reconstruction of botanical branching structure from laser scanned points.* In 11th IEEE International conference on Computer-Aided Design and Computer Graphics (CAD/Graphics 2009), 2009. (Cited on pages 65, 66 and 67.)

[Yang *et al.* 2005] Yan Yang, Lei Zhu, Steven Haker, Allen Tannenbaum and Don P. Giddens. *Harmonic Skeleton Guided Evaluation of Stenoses in Human Coronary Arteries.* In MICCAI 2005, volume 3749 of *Lecture Notes in Computer Science*, pages 490–497. Springer, 2005. (Cited on page 27.)

[Zhang 1993] Kaizhong Zhang. *A new editing based distance between unordered labeled trees.* In Proceedings of the 4th Annual Symposium on Combinatorial Pattern Matching, CPM '93, pages 254–265, London, UK, UK, 1993. Springer-Verlag. (Cited on page 97.)

[Zhang 1996] K Zhang. *A constrained edit distance between unordered labeled trees.* Algorithmica, vol. 15, no. 3, pages 205–222, 1996. (Cited on page 97.)

[Zhu *et al.* 2008] Chao Zhu, Xiaopeng Zhang, Baogang Hu and Marc Jaeger. *Reconstruction of Tree Crown Shape from Scanned Data.* In Proceedings of the 3rd international conference on Technologies for E-Learning and Digital Entertainment, Edutainment '08, pages 745–756, Berlin, Heidelberg, 2008. Springer-Verlag. (Cited on page 68.)